Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. doi: 10.18637/jss.v000.i00

Evaluating Forecasts using scoringutils in R

Nikos I. Bosse
London School of Hygiene & Tropical Medicine (LSHTM)

Hugo Gruson
LSHTM

Anne Cori
Imperial College London

Edwin van Leeuwen
UK Health Security Agency, LSHTM

Sebastian Funk
LSHTM

Sam Abbott
LSHTM

Abstract

Evaluating forecasts is essential in order to understand and improve forecasting and make forecasts useful to decision-makers. Much theoretical work has been done on the development of proper scoring rules and other scoring metrics that can help evaluate forecasts. In practice, however, conducting a forecast evaluation and comparison of different forecasters remains challenging. In this paper we introduce **scoringutils**, an R package that aims to greatly facilitate this process. It is especially geared towards comparing multiple forecasters, regardless of how forecasts were created, and visualising results. The package is able to handle missing forecasts and is the first R package to offer extensive support for forecasts represented through predictive quantiles, a format used by several collaborative ensemble forecasting efforts. The paper gives a short introduction to forecast evaluation, discusses the metrics implemented in **scoringutils** and gives guidance on when they are appropriate to use, and illustrates the application of the package using example data of forecasts for COVID-19 cases and deaths submitted to the European Forecast Hub between May and September 2021.

Keywords: forecasting, forecast evaluation, proper scoring rules, scoring, R.

1. Introduction

Good forecasts are of great interest to decision makers in various fields like finance (Timmermann 2018; Elliott and Timmermann 2016), weather predictions (Gneiting and Raftery

2005; Kukkonen *et al.* 2012) or infectious disease modeling (Reich *et al.* 2019; Funk *et al.* 2020; Cramer *et al.* 2021; Bracher *et al.* 2021b; European Covid-19 Forecast Hub 2021). Throughout the COVID-19 pandemic, forecasts from different research institutions on COVID-19 targets like reported cases and deaths have been systematically collated by several collaborative ensemble forecasting efforts ("Forecast Hubs") in the US, Germany and Poland, and Europe. An integral part of assessing and improving their usefulness is forecast evaluation. For decades, researchers have developed and refined an arsenal of techniques not only to forecast, but also to evaluate these forecasts (see e.g. Good (1952), Epstein (1969); Murphy (1971); Matheson and Winkler (1976), Gneiting, Balabdaoui, and Raftery (2007), Funk, Camacho, Kucharski, Lowe, Eggo, and Edmunds (2019), Gneiting and Raftery (2007), Bracher, Ray, Gneiting, and Reich (2021a)). Yet even with this rich body of research available, evaluating models, and in particular, comparing the performance of several models, is not trivial.

There already exist a few R (R Core Team 2021) packages which implement a wide variety of scoring metrics. The **scoringRules** package (Jordan, Krüger, and Lerch 2019a) offers a very extensive collection of functions with efficient implementations of different proper scoring rules (some of which are directly reused in **scoringutils**). **scoringutils** complements **scoringRules** in important ways, as the latter focuses on proper scoring rules only and does not implement other evaluation metrics or provide functionality to compare forecast performance visually. **scoringutils** also adds functionality to score predictive distributions that are represented by a set of quantiles and is able to handle situations with missing data. The **topmodels** package (Zeileis and Lang 2022) provides users with various graphical tools to visually evaluate and compare different forecasts. However, visualisations are only available for forecasts based on the model classes `lm`, `glm`, `crch` `disttree`, and the package is as of today not on CRAN. The **tscount** package (Liboschik, Fokianos, and Fried 2017) offers functionality to fit flexible time series models and compare the quality of the generated forecasts using different proper scoring rules. The application of these rules, however, is confined to forecasts of class `tsglm`. Other packages like **Metrics** (Hamner and Frasco 2018) and **MLmetrics** (Yan 2016) provide a collection of metrics geared towards machine learning problems, but also lack plotting functionality as well as support for a variety of metrics and tools commonly used to evaluate and compare probabilistic forecasts. In contrast to the above, **scoringutils** not only provides metrics to score individual forecasts, but simplifies the process of comparing different forecasts against each other. It accepts arbitrary forecasts regardless of how they were created by leveraging base R classes and automatically returns a variety of suitable metrics, depending on the type and format of the input forecast. It also provides functionality to facilitate comparing forecasters even when individual forecasts are missing and offers a range of plotting functions to visualise different aspects of forecast performance. The **scoringutils** package is also unique in its extensive support for forecasts in a quantile format as used by various COVID-19 Forecast Hubs (Cramer *et al.* 2021; Bracher *et al.* 2021b; European Covid-19 Forecast Hub 2021; Bracher *et al.* 2021c), which also make use of **scoringutils** for their evaluations.

The remainder of this section will provide an overview of the fundamental ideas behind forecast evaluation. Section 2 will give a detailed theoretical explanation of the evaluation metrics in **scoringutils** and when to use them. Section 3 will demonstrate how to conduct an evaluation in **scoringutils** using forecasts of COVID-19 submitted to the European Forecast Hub (European Covid-19 Forecast Hub 2021) as a case study. In the following we will use the words "model" and "forecaster" interchangeably, regardless of how forecasts were actually generated.

Table 1: Forecast and forecast target types. Forecasts can be probabilistic in nature, or a point forecast only. Depending on the type of the target (discrete, continuous or binary) different representations of the predictive distribution are possible.

Forecast type	Target type	Representation of the predictive distribution
Point forecast	continuous discrete binary	one single number for the predicted outcome
Probabilistic forecast	continuous discrete	predictive samples, closed analytical form, or quantiles
	binary	binary probabilities

1.1. Forecast types and forecast formats

In its most general sense, a forecast is the forecaster's stated belief about the future (Gneiting and Raftery 2007) that can come in many different forms. Quantitative forecasts are either point forecasts or probabilistic in nature and can make statements about continuous, discrete or binary outcome variables. Point forecasts only give one single number for the expected or most likely outcome. Probabilistic forecasts, in contrast, by definition provide a full predictive probability distribution. This makes them much more useful in any applied setting, as we learn about the forecaster's uncertainty and their belief about all aspects of the underlying data-generating distribution.

The **scoringutils** package focuses on probabilistic forecasts, and specifically on forecasts that are represented through either predictive samples or through quantiles of the predictive distributions, making it possible to evaluate arbitrary forecasts even if a closed form (i.e. parametric) distribution is not available. A variety of parametric distributions can be scored directly using **scoringRules**, but this is not yet supported in **scoringutils**.

Predictive samples offer a lot of flexibility. However, the number of samples necessary to store in order to represent the predictive distribution satisfactorily may be high. This loss of precision is usually especially pronounced in the tails of the predictive distribution. For that reason, often quantiles or central prediction intervals are reported instead. One recent example of this are the COVID-19 Forecast Hubs (Cramer *et al.* 2020, 2021; Bracher *et al.* 2021b,c; European Covid-19 Forecast Hub 2021). For binary or multinomial prediction targets, common in many classification problems, a probabilistic forecast is represented by the probability that an outcome will come true. Table 1 summarises the different forecast types and formats.

1.2. The Forecasting paradigm

Any forecaster should aim to provide a predictive distribution F that is equal to the unknown true data-generating distribution G (Gneiting *et al.* 2007). For an ideal forecast, we therefore have

$$F = G,$$

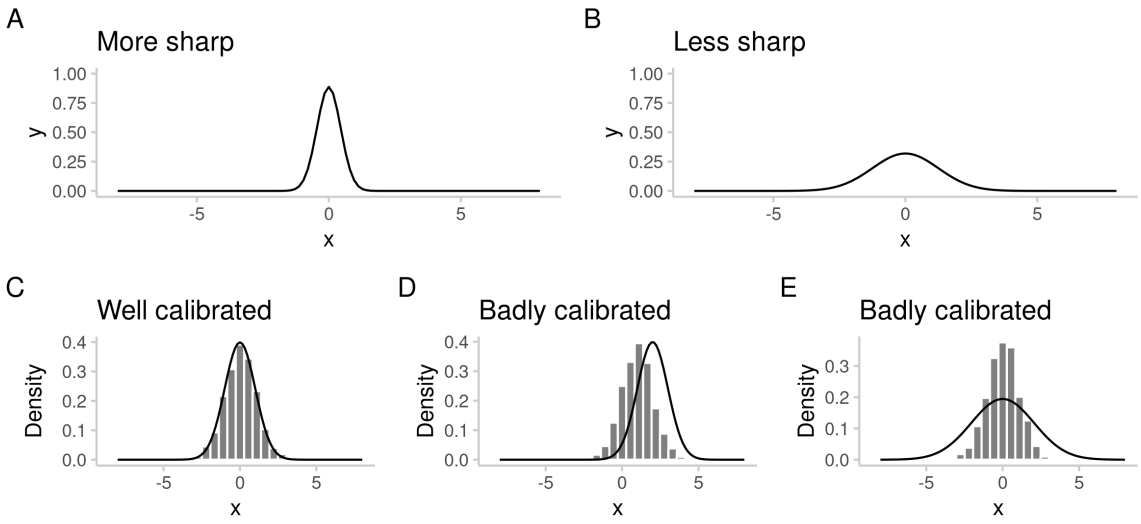

Figure 1: Schematic illustration of sharpness (A, B) and calibration (C, D, E). Sharpness is a property of the forecast (black distributions) only, while calibration is the consistency between the forecasts and the observations drawn from the true data-generating distribution (grey histograms). For illustrative purposes, the probability density function (PDF) rather than the cumulative density function (CDF) is shown.

where F and G are both cumulative distribution functions. As we don't know the true data-generating distribution G, we cannot assess the similarity between the two distributions directly. Gneiting *et al.* (2007) instead suggest to focus on two central aspects of the predictive distribution: calibration and sharpness (illustrated in Figure 1). Calibration refers to the statistical consistency (i.e. absence of systematic deviations) between the predictive distribution and the observations. One can distinguish several forms of calibration which are discussed in detail by Gneiting *et al.* (2007). Sharpness is a feature of the forecast only and describes how concentrated the predictive distribution is, i.e. how informative the forecasts are. The general forecasting paradigm states that a forecaster should maximise sharpness of the predictive distribution subject to calibration (Gneiting *et al.* 2007).

2. Scoring metrics implemented in scoringutils

An overview of the metrics implemented in **scoringutils** can be found in Table 2, while Table 4 in the Appendix provides mathematical definitions and further details. Some of the metrics in **scoringutils** focus on sharpness or calibration alone, others are so-called proper scoring rules (Gneiting and Raftery 2007), which combine both aspects into a single number. A scoring rule is proper if the ideal forecaster (i.e. one using the data-generating distribution) receives the lowest score in expectation. The scoring rule is called strictly proper, if its optimum is unique. This ensures that a forecaster evaluated by a strictly proper scoring rule is always incentivised to state their best estimate. Looking at calibration and sharpness independently can be helpful to learn about specific aspects of the forecasts and improve them. Proper scoring rules are especially useful to assess and rank predictive performance of forecasters.

Table 2: Summary table of scores available in **scoringutils**. This table (including corresponding function names) can be accessed by calling `scoringutils::metrics` in R. Not all metrics are implemented for all types of forecasts and forecasting formats, as indicated by tickmarks, '-', or '∼' (depends). D (discrete forecasts based on predictive samples), C (continuous, sample-based forecasts), B (binary), and Q (any forecasts in a quantile-based format) refer to different forecast formats. While the distinction is not clear-cut (e.g. binary is a special case of discrete), it is useful in the context of the package as available functions and functionality may differ. For a more detailed description of the terms used in this table see the corresponding paper sections (e.g. for 'global' and 'local' see section 2.3.1). For mathematical defintions of the metrics see Table 4.

Metric	D	C	B	Q	Info
Absolute error	✓	✓	–	✓	Suitable for scoring the median of a predictive distribution
Squared error	✓	✓	–	✓	Suitable for scoring the mean of a predictive distribution.
(Continuous) ranked probability score (CRPS)	✓	✓	–	–	Proper scoring rule (smaller is better), takes entire predictive distribution into account (global), penalises over− and under−confidence similarly, stable handling of outliers
Log score	–	✓	✓	–	Proper scoring rule, smaller is better, only evaluates predictive density at observed value (local), penalises over−confidence severely, susceptible to outliers
(Weighted) interval score (WIS)	✓	✓	–	✓	Proper scoring rule, smaller is better, similar properties to CRPS and converges to CRPS for an increasing number of equally spaced intervals
Dawid−Sebastiani score (DSS)	✓	✓	–	–	Proper scoring rule, smaller is better, evaluates forecast based on mean and sd of predictive distribution (global), susceptible to outliers, penalises over−confidence severely
Brier score (BS)	–	–	✓	–	Proper scoring rule, smaller is better, equals CRPS for binary outcomes, penalises over− and under−confidence similarly
Interval coverage	–	–	–	✓	Proportion of observations falling inside a given central prediction interval (= 'empirical interval coverage'). Used to assess probabilistic calibration.
Coverage deviation	–	–	–	✓	Average difference between empirical and nominal interval coverage (coverage that should have been realised)
Quantile coverage	✓	✓	–	–	Proportion of observations below a given quantile of the predictive CDF. Used to assess probabilistic calibration.
Dispersion	–	–	–	✓	Dispersion component of WIS, measures width of predictive intervals.
Median Absolute Deviation (Dispersion)	✓	✓	–	–	Measure for dispersion of a forecast: median of the absolute deviations from the median
Under−, Over−prediction	–	–	–	✓	Absolute amount of over−or under−prediction (components of WIS)
Probability integral transform (PIT)	✓	✓	–	✓	PIT transform is the CDF of the predictive distribution evaluated at the observed values. PIT values should be uniform.
Bias	✓	✓	–	✓	Measure of relative tendency to over− or under−predict (aspect of calibration), bounded between −1 and 1 (ideally 0)
Mean score ratio	∼	∼	∼	∼	Compares performance of two models. Properties depend on the metric chosen for the comparison.
(Scaled) Relative skill	∼	∼	∼	∼	Ranks models based on pairwise comparisons, useful in the context of missing forecasts. Properties depend on the metric chosen for the comparison.

2.1. Assessing calibration

There are many ways in which a forecast can be miscalibrated, i.e. systematically deviate from the observations. We also discuss metrics measuring bias, as this is an especially common form of miscalibration.

Probabilistic calibration

The form of calibration most commonly focused on is called probabilistic calibration (for other form of calibration, see Gneiting *et al.* (2007)). Probabilistic calibration means that the forecast distributions are consistent with the true data-generating distributions in the sense that on average, τ% of true observations will be below the corresponding τ-%-quantiles of the cumulative forecast distributions. This also implies that nominal coverage of the central prediction intervals (proportion of observations that should ideally be covered by the prediction intervals) corresponds to empirical coverage (proportion of observations actually covered). For example, the central 50% prediction intervals of all forecasts should really contain around 50% of the observed values, the 90% central intervals should contain around 90% of observations etc. Forecasts that are too narrow and do not cover the required proportion of observations are called overconfident or under-dispersed, while predictive distributions that are too wide are often called underconfident, over-dispersed or conservative.

One can visualise probabilistic calibration in different ways and **scoringutils** offers three options. *Interval coverage plots* (see row 3 in Figure 2) show nominal coverage of the central prediction intervals against the percentage of observed values that fall inside the corresponding prediction intervals. Ideally forecasters should lie on the diagonal line. A shift to the left means a forecaster is too conservative and issues a predictive distribution that is too wide and covers more of the observed values than needed. A shift to the right means a forecaster is overconfident and the forecast distribution is too narrow. Similarly, *quantile coverage plots* (row 4 in Figure 2) show the quantiles of the predictive distribution against the percentage of observed values below the corresponding predictive quantiles. For quantiles below the median, a line to the right of the diagonal (predictive quantiles lower than the quantiles of the data-generating distribution) means a forecaster is too conservative, while for quantiles above the median, a line to the left of the diagonal line (predictive quantiles higher than the quantiles of the data-generating distribution) implies conservative predictions.

A similar way to visualise the same information is the probability integral transform (PIT) histogram (Dawid 1984). The PIT is equal to $F(x_t)$, the cumulative predictive distribution evaluated at the observed value x_t (see more details in Table 4). If forecasts are probabilistically calibrated, then the transformed values will be uniformly distributed (for a proof see e.g. Angus (1994)). When plotting a histogram of PIT values (see row 2 in Figure 2), bias usually leads to a triangular shape, a U-shaped histogram corresponds to forecasts that are under-dispersed (too sharp) and a hump-shape appears when forecasts are over-dispersed (too wide).

It is in principle possible to formally test probabilistic calibration, for example by employing a test on the uniformity of PIT values. In practice this can be difficult as forecasts and therefore also PIT values are often correlated. We therefore advise against using formal tests in most applied settings. It is also important to note that uniformity of the PIT histogram (or a diagonal on quantile and interval coverage plots) indicates probabilistic calibration, but does not guarantee that forecasts are indeed calibrated in every relevant sense. Gneiting

et al. (2007); Hamill (2001) provide examples with different forecasters who are clearly miscalibrated, but have uniform PIT histograms.

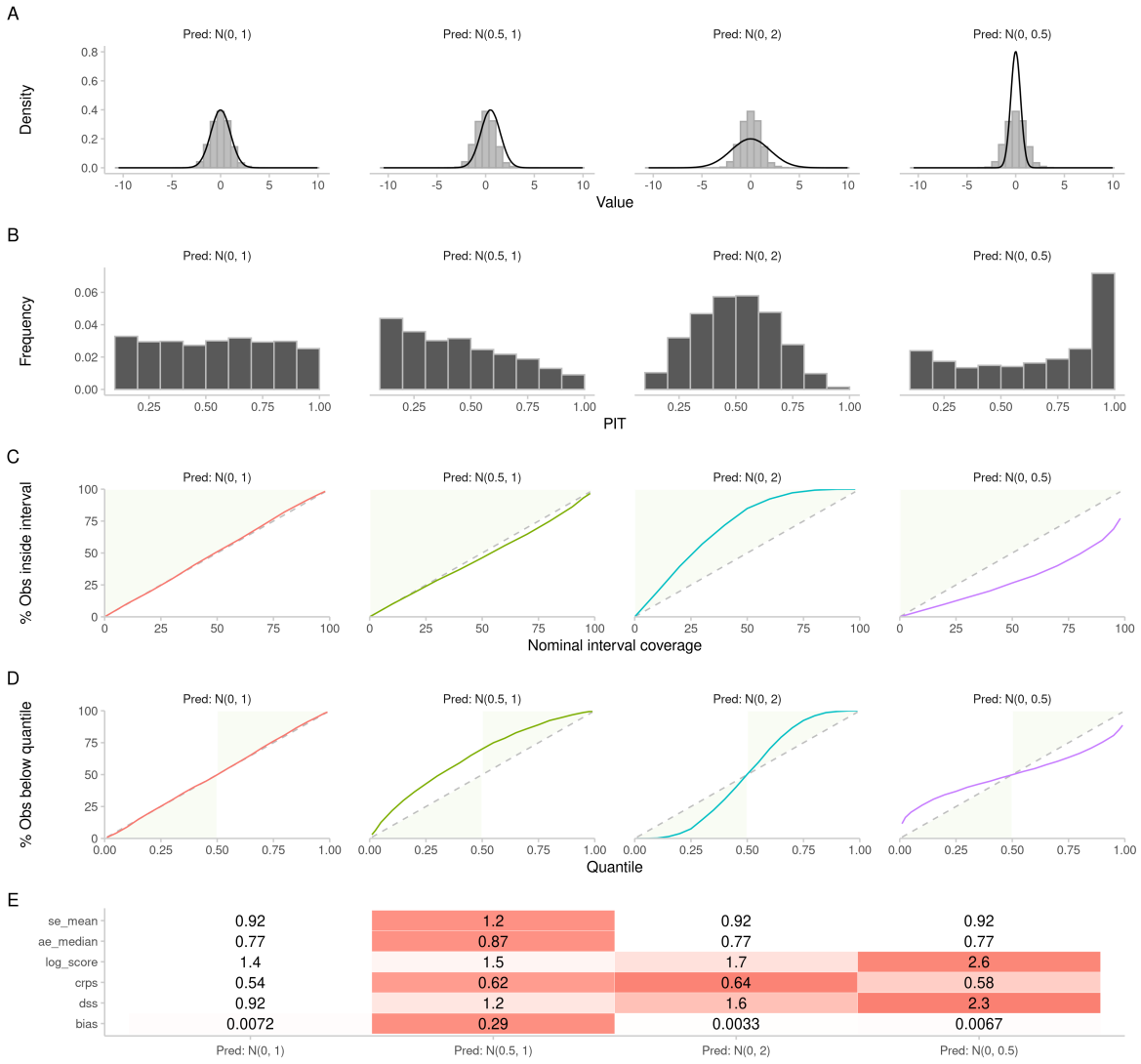

Figure 2: A: Different forecasting distributions (black) against observations sampled from a standard normal distribution (grey histograms). B: PIT histograms based on the predictive distributions and the sampled observations shown in A. C: Empirical vs. nominal coverage of the central prediction intervals for simulated observations and predictions. Areas shaded in green indicate that the forecasts are too wide (i.e. underconfident), covering more true values than they actually should, while areas in white indicate that the model generates too narrow predictions and fails to cover the desired proportion of true values with its prediction intervals. D: Quantile coverage values, with green areas indicating too wide (i.e. conservative) forecasts. E: Scores for the standard normal predictive distribution and the observations drawn from different data-generating distributions.

Bias

Biased forecasts systematically over- or under-predict the observed values. The bias metric implemented in **scoringutils** follows Funk *et al.* (2019), with slight adaptations for different forecast formats. It captures how much probability mass of the forecast was above or below the true value (mapped to values between -1 and 1, with 0 being ideal) and therefore represents a general tendency to over- or under-predict in relative terms. A value of -1 implies that the entire probability mass of the predictive distribution was below the observed value (and analogously above it for a value of 1).

For forecasts in a quantile format, bias is also reflected in the over- and underprediction components of the weighted interval score (a proper scoring rule explained in more detail in section 2.3.2). These measure over- and underprediction on an absolute scale (analogous to the absolute error of a point forecast), rather than a relative scale. However, it is not clear what the decomposition 'should' look like and a forecast can be well calibrated and still have different amounts of over- and underprediction. High overprediction or underprediction values can therefore not immediately be interpreted as systematic bias.

2.2. Assessing sharpness

Sharpness is the ability to produce narrow forecasts. In contrast to calibration it does not depend on the actual observations and is a quality of the forecast only (Gneiting *et al.* 2007). Sharpness is therefore only useful subject to calibration, as exemplified in Figure 1. For forecasts provided as samples from the predictive distribution, **scoringutils** calculates dispersion (the inverse of sharpness) as the normalised median absolute deviation (MAD), following Funk *et al.* (2019) (for details see Table 2). For quantile forecasts, we instead report the dispersion component of the weighted interval score (see details in section 2.3.2 and 4) which corresponds to a weighted average of the individual interval widths.

2.3. Proper scoring rules

Proper scoring rules for sample-based forecasts (CRPS, log score and DSS)

For forecasts in a sample format, the **scoringutils** package implements the following proper scoring rules by providing wrappers to the corresponding functions in the **scoringRules** package: the (continuous) ranked probability score (CRPS) (Epstein 1969; Murphy 1971; Matheson and Winkler 1976; Gneiting and Raftery 2007), the logarithmic score (log score) (Good 1952), and the Dawid-Sebastiani-score (DSS) (Dawid and Sebastiani 1999) (formal definitions are given in Table 4). Compared to the implementations in the **scoringRules** these are exposed to the user through a slightly adapted interface. Other, closed form variants of the CRPS, log score and DSS are available in the **scoringRules** package.

When scoring forecasts in a sample-based format, the choice is usually between the log score and the CRPS. The DSS is much less commonly used. It is easier to compute, but apart from that does not have immediate advantages over the other options. DSS, CRPS and log score differ in several important aspects: ease of estimation and speed of convergence, treatment of over- and underconfidence, sensitivity to distance Winkler, Muñoz, Cervera, Bernardo, Blattenberger, Kadane, Lindley, Murphy, Oliver, and Ríos-Insua (1996), sensitivity to outlier predictions, and sensitivity to the order of magnitude of the forecast quantity.

Estimation details and the number of samples required for accurate scoring The CRPS, DSS and log score are in principle all applicable to continuous as well as discrete forecasts. However, they differ in how easily and accurately scores can be computed based on predictive samples. This is an issue for the log score in particular, which equals the negative log density of the predictive distribution evaluated at the observed value and therefore requires a density estimation. The kernel density estimation used in **scoringutils** (through the function `log_sample()` from the **scoringRules** package) may be particularly inappropriate for discrete values (see also Table 4). The log score is therefore not computed for discrete predictions in **scoringutils**. For a small number of samples, estimated scores may deviate considerably from the exact scores computed based on closed-form predictive functions. This is especially pronounced for the log score, as illustrated in Figure 3 (adapted from (Jordan, Krüger, and Lerch 2019b)).

Overconfidence, underconfidence and outliers Proper scoring rules differ in how they penalise over- or underconfident forecasts. The log score and the DSS penalise overconfidence much more severely than underconfidence, while the CRPS does not distinguish between over- and underconfidence and penalises both rather leniently (Machete 2012) (see Figure 3B, left panel). Similarly, the CRPS is relatively lenient with regards to outlier predictions compared to the log score and the DSS (see Figure 3B, right panel). The CRPS, which can be thought of as a generalisation of the absolute error to a predictive distribution, scales linearly with the distance between forecast distribution and true value. The log score, on the other hand, as the negative logarithm of the predictive density evaluated at the observed value, can quickly tend to infinity if the probability assigned to the observed outcome is close to zero. Whether or not harsh penalisation of overconfidence and bad predictions is desirable or not depends of course on the setting. If, for example, one wanted to forecast hospital bed capacity, it may be prudent to score forecasts using a log score as one might prefer to be too cautious rather than too confident.

Sensitivity to distance - local vs. global scores The CRPS and the DSS are so-called global scoring rules, which means that the score is sensitive to the distance of the entire predictive distribution from the observed value. The log score, on the other hand, is local and the resulting score depends only on the probability density assigned to the actual outcome, ignoring the rest of the predictive distribution (see Figure 4). Sensitivity to distance (taking the entire predictive distribution into account) may be a desirable property in most settings that involve decision making. A prediction which assigns high probability to results far away from the observed value is arguably less useful than a forecast which assigns a lot of probability mass to values closer to the observed outcome (the probability assigned to the actual outcome being equal for both forecasts). The log score is only implicitly sensitive to distance in expectation if we assume that values close to the observed value are actually more likely to occur. The fact that the log score only depends on the outcome that actually realised, however, may make it more appropriate for inferential purposes (see (Winkler *et al.* 1996)) and it is commonly used in Bayesian statistics (Gelman, Hwang, and Vehtari 2014).

Sensitivity to the order of magnitude of the forecast quantity Average scores usually scale with the order of magnitude of the quantity we try to forecast (as the variance of the data-generating distribution usually increases with the mean). Figure 5 illustrates the effect

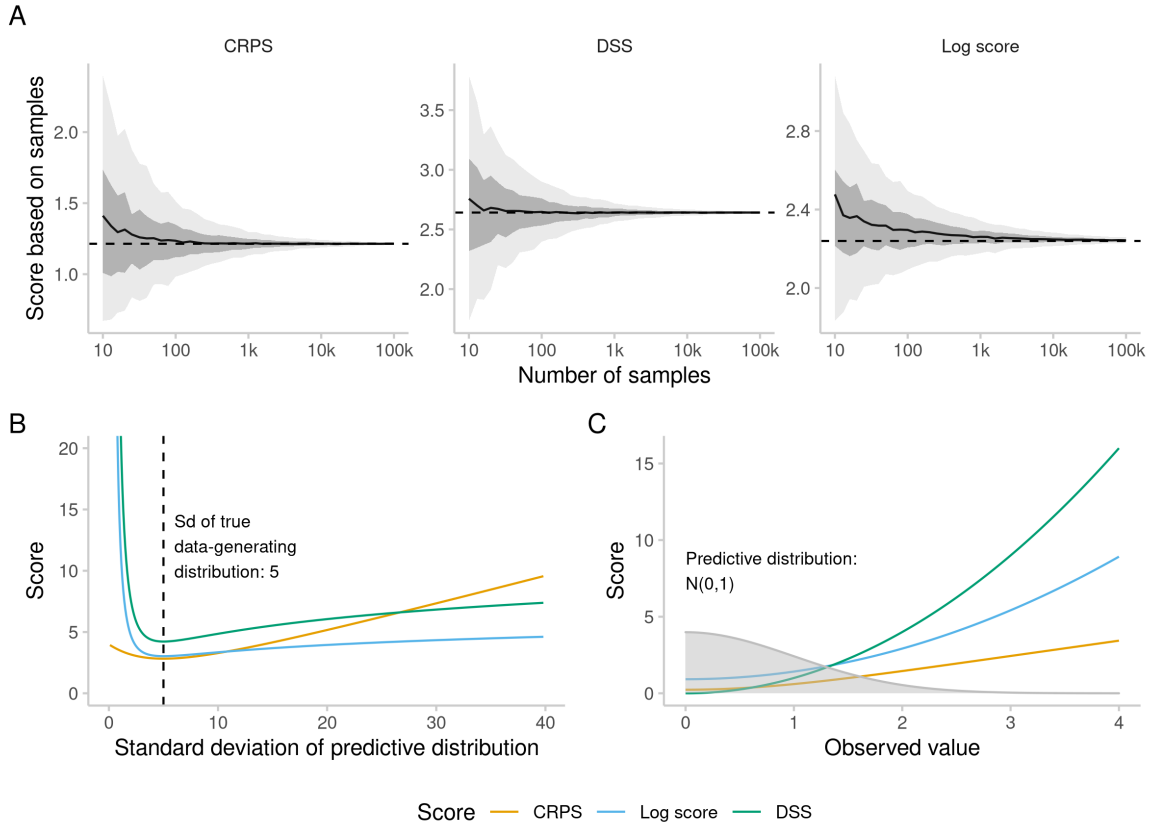

Figure 3: Top: Estimation of scores from predictive samples (adapted from (Jordan *et al.* 2019b)). Scores were computed based on samples of differing size (from 10 to 100,000). This was repeated 500 times for each sample size. The black line is the mean score across the 500 repetitions, shaded areas represent 50% and 90% intervals, and the dashed line represents the true calculated score. Bottom left: Change in score when the uncertainty of the predictive distribution is changed. The true distribution is N(0,5) with the true standard deviation marked with a dashed line, while the standard deviation of the predictive distribution is varied along the x-axis. Log score and DSS clearly punish overconfidence much more severely than underconfidence. Bottom right: Score achieved for a standard normal predictive distribution (illustrated in grey) and different true observed values. Log score and DSS punish instances more harshly where the observed value is far away from the predictive distribution.

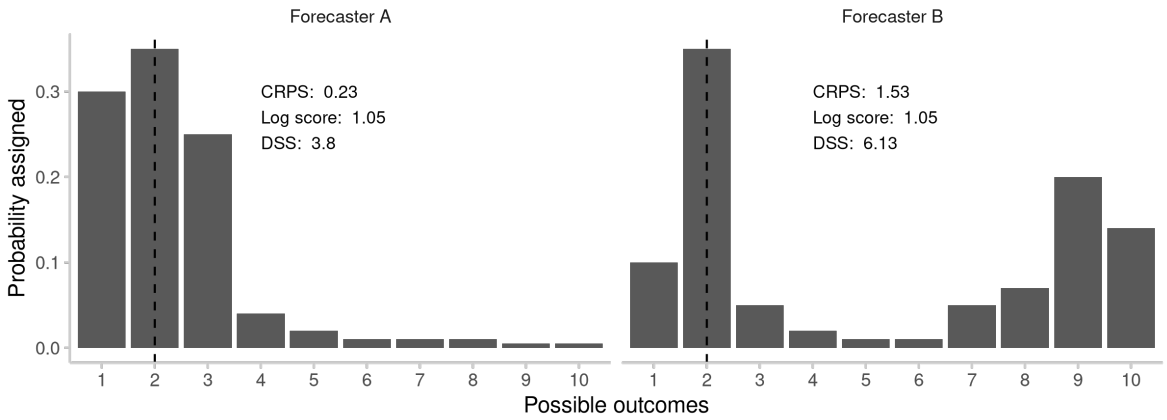

Figure 4: Probabilities assigned by two hypothetical forecasters, A and B, to the possible number of goals in a football match. The true number later observed, 2, is marked with a dashed line. Both forecasters assign a probability of 0.35 to the observed outcome, 2. Forecaster A's prediction is centred around the observed value, while Forecaster B assigns significant probability to outcomes far away from the observed value. Judged by a local score like the Log Score, both forecasters receive the same score. A global score like the CRPS and the DSS penalises forecaster B more severely.

of an increase in scale of the forecast target on average scores. This relation makes it harder to compare forecasts for very different targets, or assess average performance if the quantity of interest varies substantially over time. Average scores tend to be dominated by forecasts for targets with high absolute numbers. This is especially the case for the CRPS (as a generalisation of the absolute error), for which average scores tend to increase strongly with the order of magnitude of the quantity to forecast (see Figure 5. The log score and the DSS tend to be more robust against this effect and on average increase more slowly with an increase in the variance of the forecast target.

Proper scoring rule for quantile-based forecasts (WIS)

For forecasts in an interval or quantile format, **scoringutils** offers the weighted interval score (WIS) (Bracher *et al.* 2021a). The WIS has very similar properties to the CRPS and can be thought of as a quantile-based approximation. For an increasing number of equally-spaced prediction intervals the WIS converges to the CRPS. One additional benefit of the WIS is that it can easily be decomposed into three additive components: an uncertainty penalty (called dispersion or sharpness penalty) for the width of a prediction interval and penalties for over- and underprediction (if a value falls outside of a prediction interval).

Proper scoring rules for binary outcomes (BS and log score)

Binary forecasts can be scored using the Brier score (BS) or the log score. The Brier score (Brier 1950) corresponds to the squared difference between the given probability and the outcome (either 0 or 1) and equals the ranked probability score for the case of only two possible outcomes (Epstein 1969; Murphy 1971). The log score corresponds to the log of the probability assigned to the observed outcome. Just as with continuous forecasts, the log

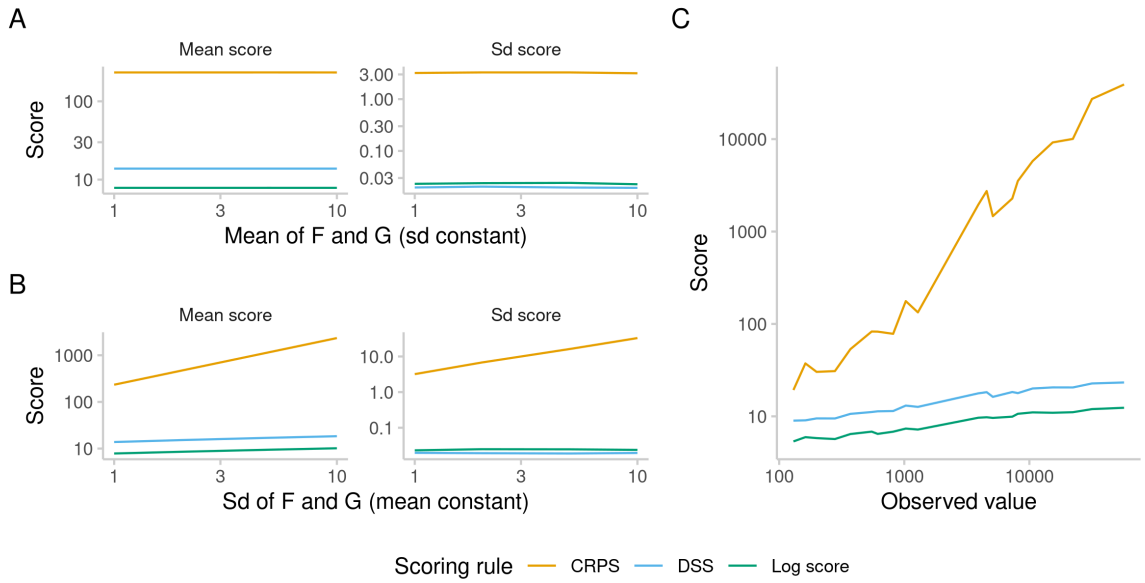

Figure 5: Scores depend on the variability of the data and therefore implicitly on the order of magnitude of the observed value. A: Mean and standard deviation of scores from a simulation of perfect forecasts with predictive distribution F equal to the true data-generating distribution G. The standard deviation of the two distributions was held constant at σ, and for different mean values μ 100 pairs of forecasts and observations were simulated. Every simulated forecast consisted of 1000 draws from the data-generating distribution G and 5000 draws from the (same) predictive distribution F. For all three scoring rules, mean and sd of the calculated scores stay constant regardless of the mean μ of F and G. B: Same setup, but now the mean of F and G was held constant at $\mu = 1$ and the standard deviation σ was varied. Average scores increase for all three scoring rules, but most strongly for the CRPS. Standard deviations of the estimated scores stay roughly constant for the DSS and log score, but also increase for the CRPS. C: Scores for forecasts of COVID-19 cases and deaths from the European Forecast Hub ensemble based on the example data provided in the package.

score penalises overconfidence much more harshly than underconfidence. The Brier score, on the other hand, penalises over- and underconfidence similarly (Machete 2012) and is more forgiving of outlier predictions.

2.4. Pairwise comparisons

Raw scores for different forecasting models are not directly comparable in the case of missing forecasts, as forecasting targets usually differ in their characteristics (e.g. the scale of the forecast target, how difficult targets are to forecast etc.). One way to mitigate this are relative skill scores based on pairwise comparisons (Cramer *et al.* 2021). Models enter a 'pairwise tournament', where all possible pairs of models are compared based on the overlapping set of available forecasts common to both models (omitting comparisons where there is no overlapping set of forecasts). For every pair, the ratio of the mean scores of both models is computed. The relative skill score of a model is then the geometric mean of all mean score ratios which involve that model. This gives us an indicator of performance relative to all other models, with the orientation depending on the score used (e.g. for the proper scoring rules presented above, a relative skill score below 1 indicates better performance). As two models can only be fairly compared if they have overlapping forecasts it is advisable to only compare forecasts that are at least 50% complete (to avoid comparisons of models that have no overlapping forecasts). Furthermore, pairwise comparisons are only possible if all scores have the same sign. One can also compute a scaled relative skill score by providing a baseline model. All individual relative skill scores are then scaled by (i.e. divided by) the relative score of the baseline model.

It is in principle possible to compute p-values to determine whether two models perform significantly differently. **scoringutils** allows to compute these using either the Wilcoxon rank sum test (also known as Mann-Whitney-U test) (Mann and Whitney 1947) or a permutation test. In practice, this is complicated by the fact that both tests assume independent observations. In reality, however, forecasts by a model may be correlated across time or another dimension (e.g. if a forecaster has a bad day, they might perform badly across different targets for a given forecast date). P-values may therefore be too liberal in suggesting significant differences where there aren't any. One way to mitigate this is to aggregate observations over a category where one suspects correlation (for example averaging across all forecasts made on a given date) before making pairwise comparisons. A test that is performed on aggregate scores will likely be more conservative.

3. Evaluating forecasts using scoringutils

This section details the core features of **scoringutils**, explains the expected data input formats and illustrates how to evaluate and compare forecasts using the example data provided in the package. **scoringutils** offers comprehensive functionality to conduct a forecast evaluation and allows users to check inputs, score forecasts and visualise results. Most functions operate on a `data.frame`-based format, but the package also provides a set of function to score individual forecasts directly which operate on vectors/matrices. These will not be discussed in this paper and we refer to the vignettes and package documentation for further information[1]. Some helper functions for data-handling, as well as example data sets and tables with additional

[1]https://epiforecasts.io/scoringutils/

information about available scoring metrics are also included in the package.

3.1. Example data

The example data included in the package and used in this paper consists of one to three week ahead forecasts made between May and September 2021 for COVID-19 cases and deaths from four different forecasting models. It represents a small subset of short-term predictions for COVID-19 cases and deaths submitted to the European Forecast Hub (European Covid-19 Forecast Hub 2021). The European Forecast Hub each week collates, aggregates and evaluates one to four week ahead predictions of different COVID-19 related targets submitted by different research groups. Forecasts are submitted in a quantile-based format with a set of 22 quantiles plus the median $(0.01, 0.025, 0.05, ..., 0.5, ...0.95, 0.975, 0.99)$. The full official hub evaluations, which also use **scoringutils**, can be seen at `https://covid19forecasthub.eu/`.

In the following, we will use the `glimpse()` function from the package **tibble** to display outputs more concisely. After loading the **scoringutils** package we can directly inspect the example data:

```
R> library(scoringutils)
R> library(tibble)
R>
R> example_quantile |>
+   na.omit() |>
+   glimpse()

Rows: 20,401
Columns: 10
$ location        <chr> "DE", "DE", "DE", "DE", "DE", "DE", "DE", "D~
$ target_end_date <date> 2021-05-08, 2021-05-08, 2021-05-08, 2021-05~
$ target_type     <chr> "Cases", "Cases", "Cases", "Cases", "Cases",~
$ true_value      <dbl> 106987, 106987, 106987, 106987, 106987, 1069~
$ location_name   <chr> "Germany", "Germany", "Germany", "Germany", ~
$ forecast_date   <date> 2021-05-03, 2021-05-03, 2021-05-03, 2021-05~
$ quantile        <dbl> 0.010, 0.025, 0.050, 0.100, 0.150, 0.200, 0.~
$ prediction      <int> 82466, 86669, 90285, 95341, 99171, 102990, 1~
$ model           <chr> "EuroCOVIDhub-ensemble", "EuroCOVIDhub-ensem~
$ horizon         <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
```

3.2. Expected input formats and data checking

Depending on the format of the forecast, a `data.frame` (or similar) is required for most functions with column names as shown in Table 3. Point forecasts are defined as forecasts that follow a quantile-based format (and can be mixed with quantile-based forecasts), but which have an `NA` value in the column `"quantile"`.

Additional columns may be present to indicate a grouping of forecasts. A combination of different columns should uniquely define the unit of a single forecast, meaning that a single forecast is defined by the combination of values in the other columns. For example, a single

Table 3: Overview of the columns required for different input formats.

Format	Required columns	Example data
quantile-based	'true_value', 'prediction', 'quantile'	example_quantile
sample-based	'true_value', 'prediction', 'sample'	example_integer, example_continuous
binary	'true_value', 'prediction'	example_binary
point-forecasts	like quantile-based, but with NA in the 'quantile' column	example_point
pairwise-comparisons	additionally a column 'model'	~

forecast could be uniquely defined by a model name, a location, a forecast date and a forecast horizon.

The function `check_forecasts()` allows to check whether input data conforms to the function requirements and returns a list with entries that provide information on what **scoringutils** infers from the data.

```
R> check_forecasts(example_quantile)

Your forecasts seem to be for a target of the following type:
$target_type
[1] "integer"

and in the following format:
$prediction_type
[1] "quantile"

The unit of a single forecast is defined by:
$forecast_unit
[1] "location"        "target_end_date" "target_type"
[4] "location_name"   "forecast_date"   "model"
[7] "horizon"

Cleaned data, rows with NA values in prediction or true_value removed:
$cleaned_data
       location target_end_date target_type true_value location_name
    1:       DE      2021-05-08       Cases     106987       Germany
    2:       DE      2021-05-08       Cases     106987       Germany
    3:       DE      2021-05-08       Cases     106987       Germany
    4:       DE      2021-05-08       Cases     106987       Germany
    5:       DE      2021-05-08       Cases     106987       Germany
   ---
20397:       IT      2021-07-24      Deaths         78         Italy
20398:       IT      2021-07-24      Deaths         78         Italy
```

```
20399:       IT      2021-07-24      Deaths         78         Italy
20400:       IT      2021-07-24      Deaths         78         Italy
20401:       IT      2021-07-24      Deaths         78         Italy
       forecast_date quantile prediction                model
    1:    2021-05-03    0.010      82466 EuroCOVIDhub-ensemble
    2:    2021-05-03    0.025      86669 EuroCOVIDhub-ensemble
    3:    2021-05-03    0.050      90285 EuroCOVIDhub-ensemble
    4:    2021-05-03    0.100      95341 EuroCOVIDhub-ensemble
    5:    2021-05-03    0.150      99171 EuroCOVIDhub-ensemble
   ---
20397:    2021-07-12    0.850        352  epiforecasts-EpiNow2
20398:    2021-07-12    0.900        397  epiforecasts-EpiNow2
20399:    2021-07-12    0.950        499  epiforecasts-EpiNow2
20400:    2021-07-12    0.975        611  epiforecasts-EpiNow2
20401:    2021-07-12    0.990        719  epiforecasts-EpiNow2
       horizon
    1:       1
    2:       1
    3:       1
    4:       1
    5:       1
   ---
20397:       2
20398:       2
20399:       2
20400:       2
20401:       2

Number of unique values per column per model:
$unique_values
                   model location target_end_date target_type
1: EuroCOVIDhub-ensemble        4              12           2
2: EuroCOVIDhub-baseline        4              12           2
3:  epiforecasts-EpiNow2        4              12           2
4:       UMass-MechBayes        4              12           1
   true_value location_name forecast_date quantile prediction horizon
1:         96             4            11       23       3969       3
2:         96             4            11       23       3733       3
3:         95             4            11       23       3903       3
4:         48             4            11       23       1058       3

$messages
[1] "144 values for `prediction` are NA in the data provided and the corresponding rows we
```

The values stored in the list elements `target_type` and `prediction_type` refer to type of the forecast and the target variable. `forecast_unit` contains a vector of the columns which **scoringutils** thinks denote the unit of a single forecast. This means that in this instance a

single forecast (with a set of 23 quantiles) can uniquely be identified by the values in the columns "location", "target_end_date", "target_type", "location_name", "forecast_date", "model", "horizon". In this example, having "location" as well as "location_name" included does not make a difference, as they contain duplicated information. In general, however, it is strongly advised to remove all unnecessary columns that do not help identify a single forecast. `unique_values` gives an overview of the number of unique values per column across the entire data set, providing a first hint as to whether the forecast set is complete. `messages` or `warnings` show messages and warnings created when checking the data.

3.3. Visualising forecast data

It is helpful to start the evaluation process by examining forecast availability, as missing forecasts can impact the evaluation if missingness correlates with performance. The function `avail_forecasts()` returns information about the number of available forecasts, given a level of summary that can be specified through the `by` argument. For example, to see how many forecasts there are per model and target_type, we can run

```
R> avail_forecasts(data = example_integer,
+                  by = c("model", "target_type"))

                    model target_type Number forecasts
1: EuroCOVIDhub-ensemble       Cases              128
2: EuroCOVIDhub-baseline       Cases              128
3:  epiforecasts-EpiNow2       Cases              128
4: EuroCOVIDhub-ensemble      Deaths              128
5: EuroCOVIDhub-baseline      Deaths              128
6:       UMass-MechBayes      Deaths              128
7:  epiforecasts-EpiNow2      Deaths              119
```

and visualise results using the function `plot_avail_forecasts()`. The plot resulting from running the following code is displayed in Figure 6.

```
R> library(ggplot2)
R>
R> avail_forecasts(data = example_integer,
+                  by = c("model", "target_type", "forecast_date")) |>
+   plot_avail_forecasts(x = "forecast_date",
+                        show_numbers = FALSE) +
+   facet_wrap(~ target_type) +
+   labs(y = "Model", x = "Forecast date")
```

The forecasts and observed values themselves can be visualised using the `plot_predictions()` function and its `make_na()` helper function. `make_na()` represents a form of filtering, but instead of filtering entire rows, the relevant entries in the columns "prediction" or "true_value" are made `NA`. This allows the user to filter observations and forecasts independently. In order to be able to facet the plot correctly, `plot_predictions()` has a an additional `by` argument in which the user needs to specify all columns relevant for facetting. In order to be

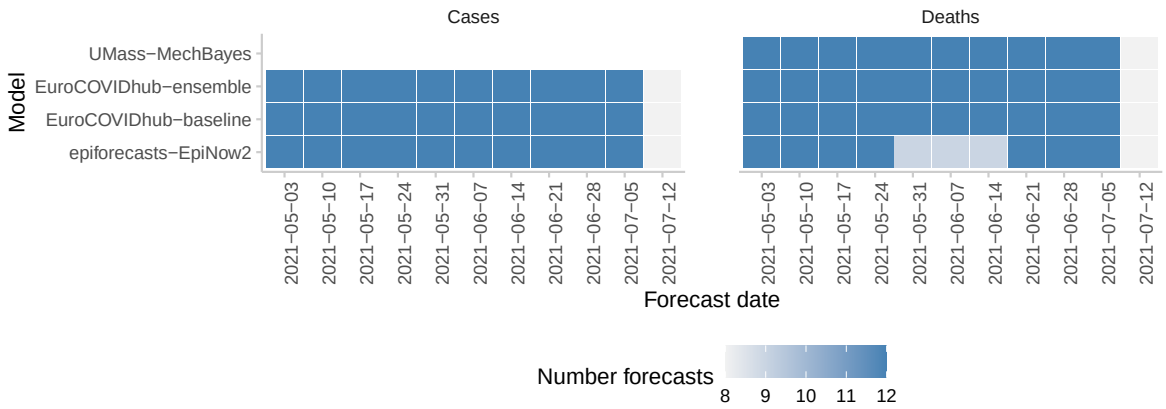

Figure 6: Overview of the number of available forecasts.

To display, for example, short-term forecasts for COVID-19 cases and deaths made by the EuroCOVIDhub-ensemble model on June 28 2021 as well as 5 weeks of prior data, we can call the following. The resulting plot is shown in Figure 7.

```
R> example_quantile %>%
+   make_na(what = "truth",
+           target_end_date > "2021-07-15",
+           target_end_date <= "2021-05-22") %>%
+   make_na(what = "forecast",
+           model != "EuroCOVIDhub-ensemble",
+           forecast_date != "2021-06-28") %>%
+   plot_predictions(x = "target_end_date", by = c("target_type", "location")) +
+   aes(colour = model, fill = model) +
+   facet_wrap(target_type ~ location, ncol = 4, scales = "free_y") +
+   labs(x = "Target end date")
```

3.4. Scoring forecasts with `score()`

The function `score()` evaluates predictions against observed values and automatically applies the appropriate scoring metrics depending on the input data.

We can simply call:

```
R> score(example_quantile) |>
+   glimpse()

Rows: 20,401
Columns: 18
$ location           <chr> "DE", "DE", "DE", "DE", "DE", "DE", "DE",~
$ target_end_date    <date> 2021-05-08, 2021-05-08, 2021-05-08, 2021~
$ target_type        <chr> "Cases", "Cases", "Cases", "Cases", "Case~
```

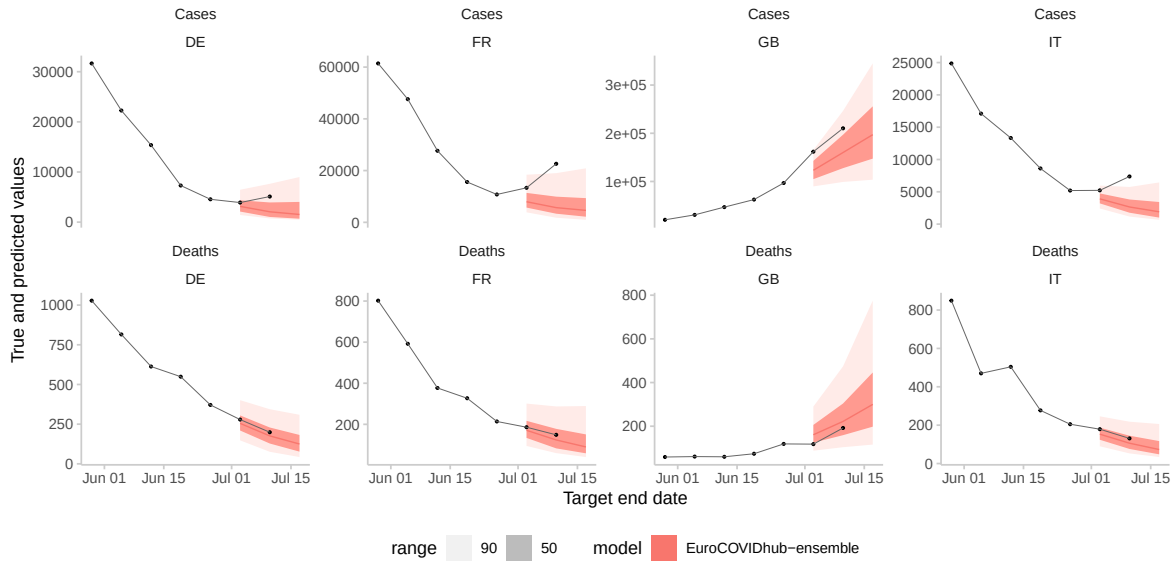

Figure 7: Short-term forecasts for COVID-19 cases and deaths made by the EuroCOVIDhub-ensemble model on June 28 2021.

```
$ location_name      <chr> "Germany", "Germany", "Germany", "Germany~
$ forecast_date      <date> 2021-05-03, 2021-05-03, 2021-05-03, 2021~
$ model              <chr> "EuroCOVIDhub-baseline", "EuroCOVIDhub-ba~
$ horizon            <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
$ range              <dbl> 0, 10, 10, 20, 20, 30, 30, 40, 40, 50, 50~
$ interval_score     <dbl> 25620.00, 25599.50, 25599.50, 25481.00, 2~
$ dispersion         <dbl> 0.00, 184.50, 184.50, 556.00, 556.00, 816~
$ underprediction    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
$ overprediction     <dbl> 25620, 25415, 25415, 24925, 24925, 24454,~
$ coverage           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
$ coverage_deviation <dbl> 0.00, -0.10, -0.10, -0.20, -0.20, -0.30, ~
$ bias               <dbl> 0.95, 0.95, 0.95, 0.95, 0.95, 0.95, 0.95,~
$ quantile           <dbl> 0.500, 0.450, 0.550, 0.400, 0.600, 0.350,~
$ ae_median          <dbl> 25620, 25620, 25620, 25620, 25620, 25620,~
$ quantile_coverage  <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,~
```

The above produces one score for every forecast. However, we usually like to summarise scores to learn about average performance across certain categories. This can be done using the function `summarise_scores()`, which returns one summarised score per category (column name) specified in the argument `by`. To return, for example, one score per model and forecast target, we can run the following:

```
R> score(example_quantile) |>
+   summarise_scores(by = c("model", "target_type")) |>
+   glimpse()

Rows: 7
```

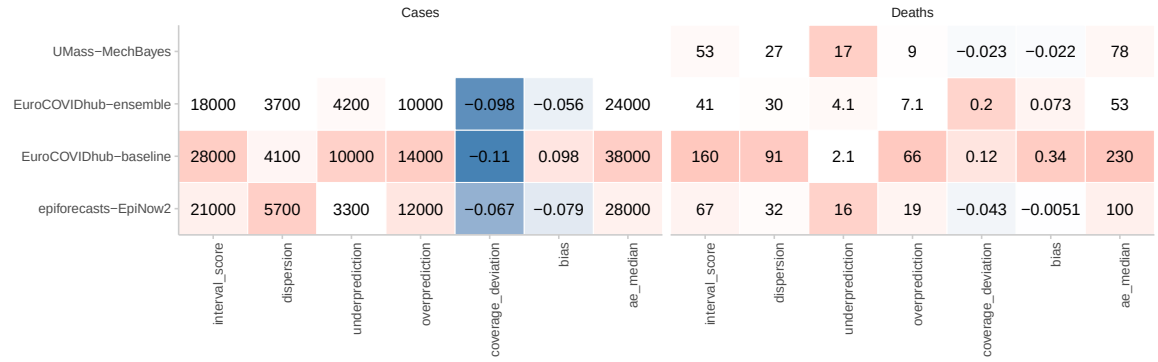

Figure 8: Coloured table to visualise the computed scores. Red colours indicate that a value is higher than ideal, blue indicates it is lower than ideal and the opacity indicates the strength of the deviation from the ideal.

```
Columns: 9
$ model              <chr> "EuroCOVIDhub-baseline", "EuroCOVIDhub-en~
$ target_type        <chr> "Cases", "Cases", "Cases", "Deaths", "Dea~
$ interval_score     <dbl> 28483.57465, 17943.82383, 20831.55662, 15~
$ dispersion         <dbl> 4102.50094, 3663.52458, 5664.37795, 91.40~
$ underprediction    <dbl> 10284.972826, 4237.177310, 3260.355639, 2~
$ overprediction     <dbl> 14096.100883, 10043.121943, 11906.823030,~
$ coverage_deviation <dbl> -0.11211957, -0.09785326, -0.06660326, 0.~
$ bias               <dbl> 0.09796875, -0.05640625, -0.07890625, 0.3~
$ ae_median          <dbl> 38473.60156, 24101.07031, 27923.81250, 23~
```

Summarised scores can then be visualised using the function `scores_table()`. In order to display scores it is often useful to round the output to e.g. two significant digits, which can be achieved through another call of `summarise_scores()`. The output of the following is shown in Figure 8:

```
R> score(example_quantile) |>
+   summarise_scores(by = c("model", "target_type")) |>
+   summarise_scores(fun = signif, digits = 2) |>
+   plot_score_table(y = "model", by = "target_type") +
+   facet_wrap(~ target_type)
```

While `summarise_scores()` accepts arbitrary summary functions, care has to be taken when using something else than `mean()`, because scores may lose propriety when using other summary functions. For example, the median of several individual scores (individually based on a proper scoring rule) is usually not proper. A forecaster judged by the median of several scores may be incentivised to misrepresent their true belief in a way that is not true for the mean score.

The user must exercise additional caution and should usually avoid aggregating scores across categories which differ much in the magnitude of the quantity to forecast, as forecast errors

usually increase with the order of magnitude of the forecast target. In the given example, looking at one score per model (i.e. specifying `summarise_by = c("model")`) is problematic, as overall aggregate scores would be dominated by case forecasts, while performance on deaths would have little influence. Similarly, aggregating over different forecast horizons is often ill-advised as the mean will be dominated by further ahead forecast horizons. In some instances it may be helpful to look at relative skill scores instead (see sections 2.4 and 3.5).

As a proxy for calibration, we are often interested in empirical coverage-levels of certain central prediction intervals, for example the percentage of true values which fell inside all 50% or 90% prediction intervals. For any quantile-based forecast, we can simply add this information using the function `add_coverage()`. The function has a `by` argument which accepts a vector of column names defining the level of grouping for which empirical coverage is computed. Note that these column names should be equal to those passed to `by` in subsequent calls of `summarise_forecasts()`.

For sample-based forecasts, calculating coverage requires an extra step, namely estimating quantiles of the predictive distribution from samples. The function `sample_to_quantile()` takes a `data.frame` in a sample-based format and outputs one in a quantile-based format, which can then be passed to `score()` and `add_coverage()`:

```
R> q <- c(0.01, 0.025, seq(0.05, 0.95, 0.05), 0.975, 0.99)
R>
R> example_integer |>
+   sample_to_quantile(quantiles = q) |>
+   score() |>
+   add_coverage(ranges = c(50, 90), by = c("model", "target_type")) |>
+   summarise_scores(by = c("model", "target_type")) |>
+   glimpse()

Rows: 7
Columns: 11
$ model              <chr> "EuroCOVIDhub-baseline", "EuroCOVIDhub-ba~
$ target_type        <chr> "Cases", "Deaths", "Cases", "Deaths", "De~
$ interval_score     <dbl> 28516.90313, 147.02220, 18338.07715, 45.0~
$ dispersion         <dbl> 4743.98408, 93.75104, 3756.73019, 30.5956~
$ underprediction    <dbl> 10896.738488, 1.589368, 4379.957812, 4.86~
$ overprediction     <dbl> 12876.180564, 51.681793, 10201.389147, 9.~
$ coverage_deviation <dbl> -0.10600543, 0.08896739, -0.11959239, 0.1~
$ bias               <dbl> 0.03296875, 0.35898438, -0.04835938, 0.07~
$ ae_median          <dbl> 37648.04297, 217.01562, 24749.40625, 62.6~
$ coverage_50        <dbl> 0.3828125, 0.6406250, 0.3515625, 0.828125~
$ coverage_90        <dbl> 0.7265625, 0.9921875, 0.8046875, 0.984375~
```

The process is designed to require conscious action by the user, because the estimation of quantiles from predictive samples may be biased if the number of available samples is not sufficiently large.

3.5. Pairwise comparisons

In order to obtain a model ranking, we recommend looking at the relative skill in terms of an appropriate proper scoring rule instead of the raw score whenever forecasts are missing. Relative skill scores can either be obtained by specifying `relative_skill = TRUE` in the function `summarise_scores()`, or by calling the function `pairiwse_comparison()`. In both cases, pairwise comparisons are computed according to the grouping specified in the argument `by`: internally, the `data.frame` with all scores gets split into different `data.frame`s according to the values specified in `by` (excluding the column 'model'). Relative scores are then computed for every individual group separately. In the example below we specify `by = c("model", "target_type")`, which means that there is one relative skill score per model, calculated separately for the different forecasting targets. Using the argument `baseline`, we can compute relative skill with respect to a baseline model.

```
R> score(example_quantile) |>
+   pairwise_comparison(by = c("model", "target_type"),
+                       baseline = "EuroCOVIDhub-baseline") |>
+   glimpse()

Rows: 25
Columns: 8
$ model             <chr> "EuroCOVIDhub-baseline", "EuroCOVIDhub-bas~
$ target_type       <chr> "Cases", "Cases", "Cases", "Cases", "Cases~
$ compare_against   <chr> "epiforecasts-EpiNow2", "EuroCOVIDhub-ense~
$ mean_scores_ratio <dbl> 1.3673282, 1.5873748, 1.0000000, 0.8613770~
$ pval              <dbl> 1.824256e-08, 2.953792e-17, 1.000000e+00, ~
$ adj_pval          <dbl> 3.648512e-08, 8.861377e-17, 1.000000e+00, ~
$ relative_skill    <dbl> 1.2947445, 1.2947445, 1.2947445, 0.8156514~
$ scaled_rel_skill  <dbl> 1.0000000, 1.0000000, 1.0000000, 0.6299709~
```

Pairwise comparisons should usually be made based on unsummarised scores (the function `pairwise_comparison()` internally summarises over samples and quantiles automatically, but nothing else), as summarising can change the set of overlapping forecasts between two models and distort relative skill scores. When using `pairwise_comparison()`, the function `summarise_scores()` should therefore usually not be called beforehand. One potential exception to this is when one is interested in the p-values obtained from pairwise comparisons. As forecasts are usually highly correlated (which the calculation of p-values do not account for), it may be sensible to summaries over a few categories (provided there are no missing values within the categories summarised over) to reduce correlation and obtain more conservative p-values.

Using the function `plot_pairwise_comparison()` we can visualise the mean score ratios between all models. The output of the following code is shown in Figure 9.

```
R> score(example_quantile) |>
+   pairwise_comparison(by = c("model", "target_type"),
+                       baseline = "EuroCOVIDhub-baseline") |>
+   plot_pairwise_comparison() +
+   facet_wrap(~ target_type)
```

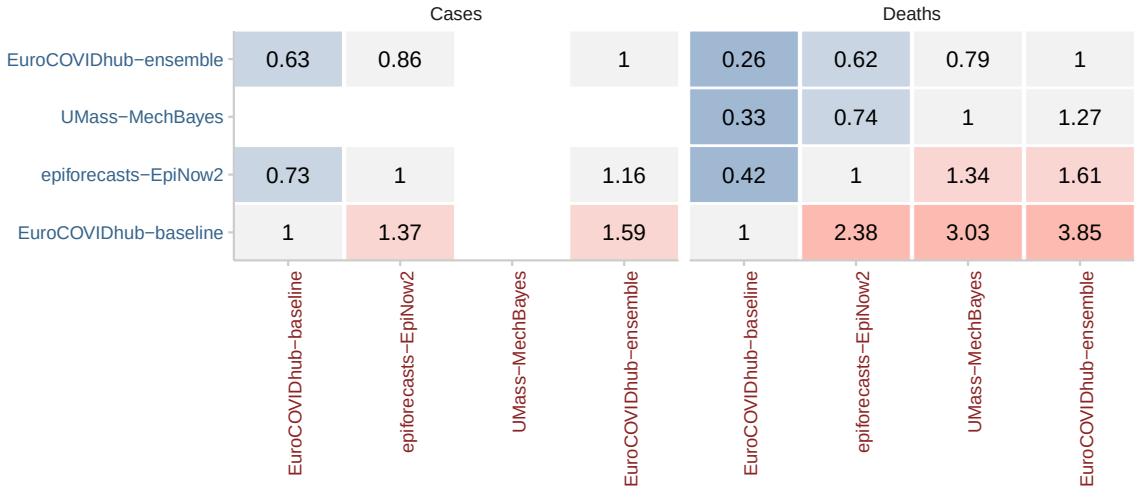

Figure 9: Ratios of mean scores based on overlapping forecast sets. When interpreting the plot one should look at the model on the y-axis, and the model on the x-axis is the one it is compared against. If a tile is blue, then the model on the y-axis performed better. If it is red, the model on the x-axis performed better in direct comparison. In the example above, the EuroCOVIDhub-ensemble performs best (it only has values smaller than one), while the EuroCOVIDhub-baseline performs worst (and only has values larger than one). For cases, the UMass-MechBayes model is excluded as there are no case forecasts available and therefore the set of overlapping forecasts is empty.

3.6. Model diagnostics

The **scoringutils** package offers a variety of functions to aid the user in diagnosing issues with models. For example, to detect systematic patterns it may be useful to visualise a single metric across several dimensions. The following produces a heatmap of bias values across different locations and forecast targets (output shown in Figure 10).

```
R> score(example_continuous) |>
+   summarise_scores(by = c("model", "location", "target_type")) |>
+   plot_heatmap(x = "location", metric = "bias") +
+     facet_wrap(~ target_type)
```

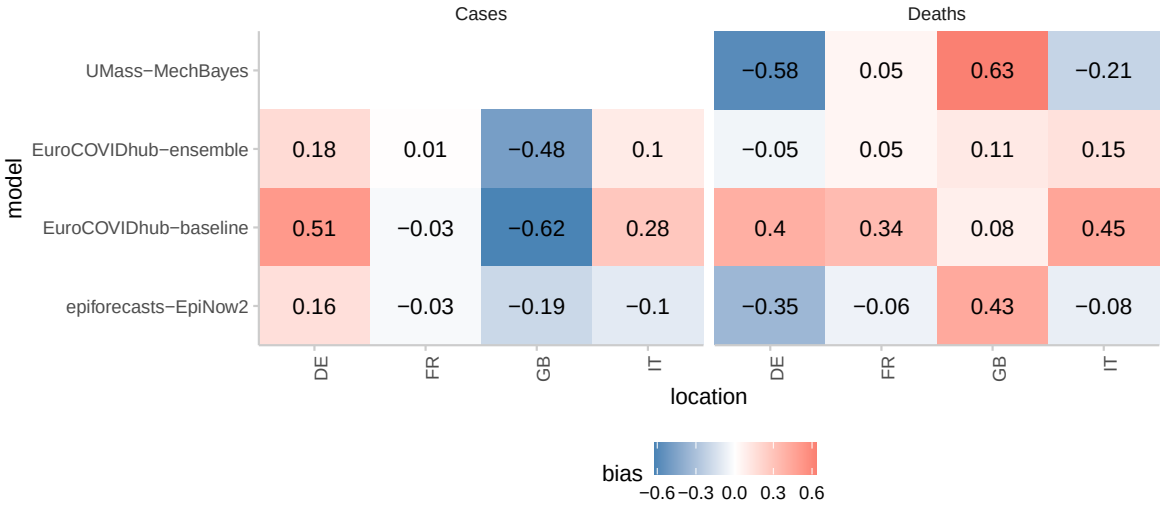

Figure 10: Heatmap of bias values for different models across different locations and forecast targets. Bias values are bound between -1 (underprediction) and 1 (overprediction) and should be 0 ideally. Red tiles indicate an upwards bias (overprediction), while blue tiles indicate a downwards bias (under-predicction)

For quantile-based forecasts, it is helpful to visualise the decomposition of the weighted interval score into its components: dispersion, overprediction and underprediction. This can be achieved using the function `plot_wis()`, as shown in Figure 11

```
R> score(example_quantile) |>
+   summarise_scores(by = c("model", "target_type")) |>
+   plot_wis(relative_contributions = FALSE) +
+   facet_wrap(~ target_type,
+              scales = "free_x")
```

Special attention should be given to calibration. The most common way of assessing calibration (more precisely: probabilistic calibration) are PIT histograms, as explained in section 2.1.1. Ideally, PIT values should be uniformly distributed after the transformation.

We can compute PIT values in the following way:

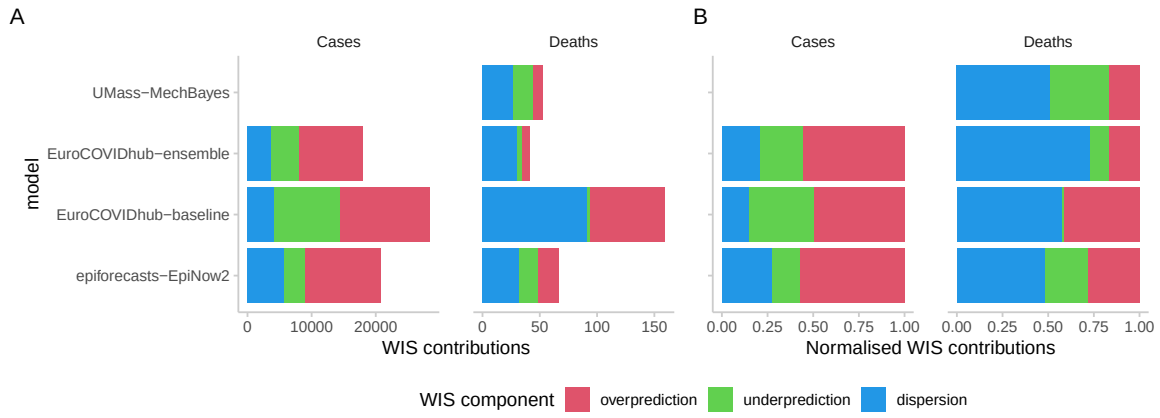

Figure 11: Decomposition of the weighted interval score (WIS) into dispersion, overprediction and underprediction. A: absolute contributions, B: contributions normalised to 1.

```
R> example_continuous |>
+   pit(by = "model")
```

```
                    model pit_value
  1: EuroCOVIDhub-baseline     0.025
  2: EuroCOVIDhub-baseline     0.525
  3: EuroCOVIDhub-baseline     0.000
  4: EuroCOVIDhub-baseline     0.000
  5: EuroCOVIDhub-baseline     0.200
 ---
883:       UMass-MechBayes     0.950
884:       UMass-MechBayes     0.500
885:       UMass-MechBayes     0.100
886:       UMass-MechBayes     0.450
887:       UMass-MechBayes     0.100
```

and create PIT histograms using the function `plot_pit()`. The output of the following is shown in Figure 12:

```
R> example_continuous |>
+   pit(by = c("model", "target_type")) |>
+   plot_pit() +
+   facet_grid(target_type ~ model)
```

We can also look at interval and quantile coverage plots (explained in more detail in section 2.1.1) using the functions `plot_interval_coverage()` and `plot_quantile_coverage()`. These plots require that the columns "range" and "quantile", respectively, be present in the scores to plot, and therefore need to be included in the `by` argument when summarising scores. The output of the following is shown in Figure 13.

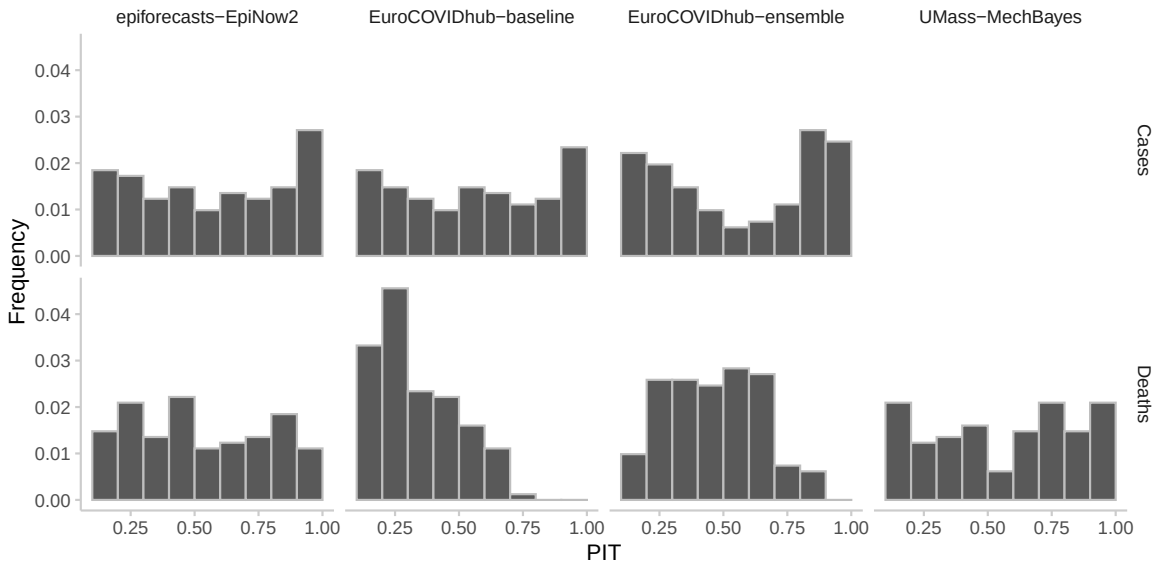

Figure 12: PIT histograms of all models stratified by forecast target. Histograms should ideally be uniform. A u-shape usually indicates overconfidence (forecasts are too narrow), a hump-shaped form indicates underconfidence (forecasts are too uncertain) and a triangle-shape indicates bias.

```
R> cov_scores <- score(example_quantile) |>
+   summarise_scores(by = c("model", "target_type", "range", "quantile"))
R>
R> plot_interval_coverage(cov_scores) +
+   facet_wrap(~ target_type)
R>
R> plot_quantile_coverage(cov_scores) +
+   facet_wrap(~ target_type)
```

It may sometimes be interesting to see how different scores correlate with each other. We can examine this using the function `correlation()`. When dealing with quantile-based forecasts, it is important to call `summarise_scores()` before `correlation()` in order to summarise over quantiles before computing correlations. The plot resulting from the following code is shown in Figure 14.

```
R> correlations <- example_quantile |>
+   score() |>
+   summarise_scores() |>
+   correlation()
R>
R> correlations |>
+   glimpse()

Rows: 7
Columns: 8
```

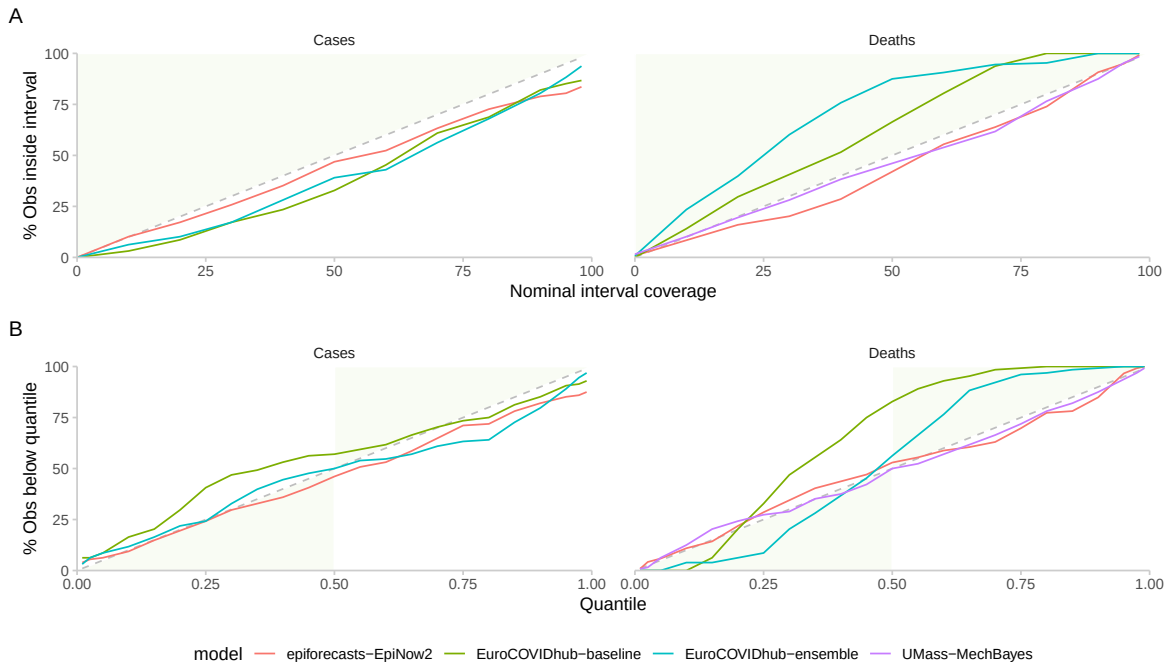

Figure 13: Interval coverage (A) and quantile coverage (B) plots. Areas shaded in green indicate that the forecasts are too wide (i.e. underconfident), while areas in white indicate that the model is overconfident and generates too narrow predictions intervals.

```
$ interval_score     <dbl> 1.00, 0.46, 0.28, 0.94, -0.34, 0.11, 0.99
$ dispersion         <dbl> 0.46, 1.00, 0.15, 0.32, -0.12, 0.11, 0.54
$ underprediction    <dbl> 0.28, 0.15, 1.00, -0.03, -0.33, -0.35, 0.~
$ overprediction     <dbl> 0.94, 0.32, -0.03, 1.00, -0.25, 0.22, 0.90
$ coverage_deviation <dbl> -0.34, -0.12, -0.33, -0.25, 1.00, 0.06, -~
$ bias               <dbl> 0.11, 0.11, -0.35, 0.22, 0.06, 1.00, 0.10
$ ae_median          <dbl> 0.99, 0.54, 0.34, 0.90, -0.38, 0.10, 1.00
$ metric             <chr> "interval_score", "dispersion", "underpre~
```

```
R> correlations |>
+   plot_correlation()
```

3.7. Summary and discussion

Forecast evaluation is invaluable to understanding and improving current forecasts. The **scoringutils** package aims to facilitate this process and make it easier, even for less experienced users. It provides a fast, flexible and convenient evaluation framework based on `data.frame`s, but also makes a set of scoring functions available to more experienced users to be used in other packages or pipelines. A set of visualisations and plotting functions help with diagnosing issues with models and allow for thorough comparison between different forecasting approaches.

The package is still under active development and we warmly welcome contributions to **scoringutils**. In the future we hope to extend the number of scoring metrics supported.

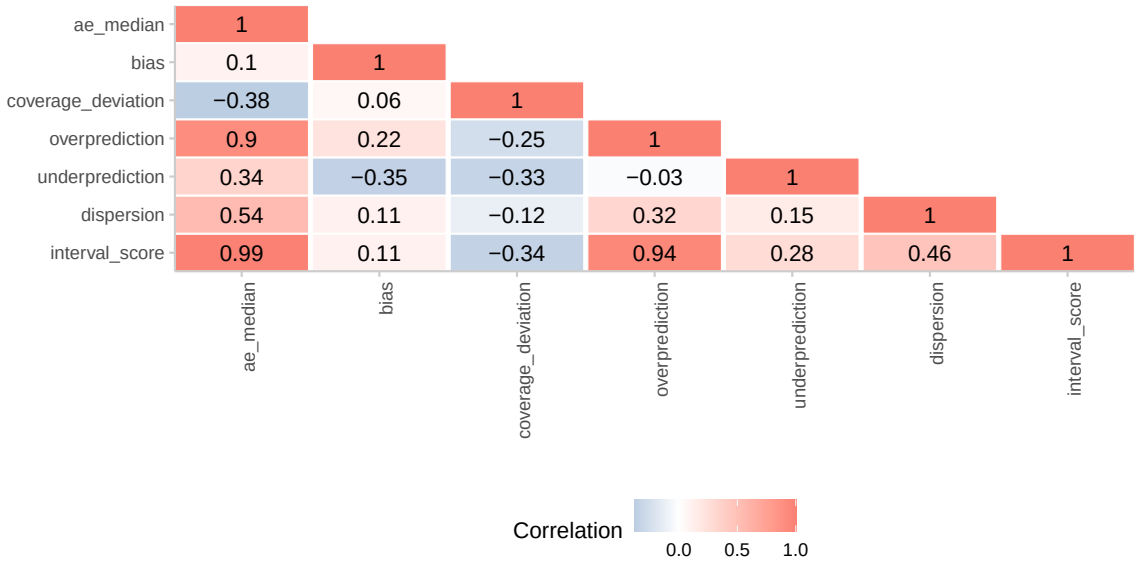

Figure 14: Correlation between different scores

This includes spherical scoring rules (Gneiting and Raftery 2007; Jose 2009; Machete 2012), evaluation of multinomial prediction tasks, as well as a broader range of scoring metrics for point forecasts. We also plan to expand the plotting functionality and hope to make templates available for automated scoring reports.

3.8. Acknowledgments

Funding statements

NIB received funding from the Health Protection Research Unit (grant code NIHR200908). HG MISSING. AC acknowledges funding by the NIHR, the Sergei Brin foundation, USAID, and the Academy of Medical Sciences. EvL acknowledges funding by the National Institute for Health Research (NIHR) Health Protection Research Unit (HPRU) in Modelling and Health Economics (grant number NIHR200908) and the European Union's Horizon 2020 research and innovation programme - project EpiPose (101003688). SF's work was supported by the Wellcome Trust (grant: 210758/Z/18/Z), and the NIHR (NIHR200908). SA's work was funded by the Wellcome Trust (grant: 210758/Z/18/Z). This study is partially funded by the National Institute for Health Research (NIHR) Health Protection Research Unit in Modelling and Health Economics, a partnership between UK Health Security Agency and Imperial College London in collaboration with LSHTM (grant code NIHR200908); and acknowledges funding from the MRC Centre for Global Infectious Disease Analysis (reference MR/R015600/1), jointly funded by the UK Medical Research Council (MRC) and the UK Foreign, Commonwealth & Development Office (FCDO), under the MRC/FCDO Concordat agreement and is also part of the EDCTP2 programme supported by the European Union. Disclaimer: "The views expressed are those of the author(s) and not necessarily those of the NIHR, UKHSA or the Department of Health and Social Care. We thank Community Jameel for Institute and research funding

(APPENDIX) Detailed Information on Metrics

Table 4: Detailed explanation of all the metrics,

Metric	Explanation
CRPS (Continuous) ranked probability score	The crps is a proper scoring rule that generalises the absolute error to probabilistic forecasts. It measures the 'distance' of the predictive distribution to the observed data-generating distribution. The CRPS is given as $$\text{CRPS}(F, y) = \int_{-\infty}^{\infty} (F(x) - 1(x \geq y))^2 \, dx,$$ where y is the true observed value and F the CDF of predictive distribution. Often An alternative representation is used: $$\text{CRPS}(F, y) = \frac{1}{2}\mathbb{E}_F\lvert X - X'\rvert - \mathbb{E}_P\lvert X - y\rvert,$$ where X and X' are independent realisations from the predictive distributions F with finite first moment and y is the true value. In this represenation we can simply replace X and X' by samples sum over all possible combinations to obtain the CRPS. For integer-valued forecasts, the RPS is given as $$\text{RPS}(F, y) = \sum_{x=0}^{\infty} (F(x) - 1(x \geq y))^2.$$ **Usage and caveats** Smaller values are better. The crps is a good choice for most practical purposes that involve decision making, as it takes the entire predictive distribution into account. If two forecasters assign the same probability to the true event y, then the forecaster who assigned high probability to events far away from y will still get a worse score. The crps (in contrast to the log score) can at times be quite lenient towards extreme mispredictions. Also, due to it's similarity to the absolute error, the level of scores depend a lot on the absolute value of what is predicted, which makes it hard to compare scores of forecasts for quantities that are orders of magnitude apart.

Table 4: Detailed explanation of all the metrics, *(continued)*

Metric	Explanation
Log score	The Log score is a proper scoring rule that is simply compuated as the log of the predictive density evaluated at the true observed value. It is given as $$\text{log score} = \log f(y),$$ where f is the predictive density function and y is the true value. For integer-valued forecasts, the log score can be computed as $$\text{log score} = \log p_y,$$ where p_y is the probability assigned to outcome p by the forecast F. **Usage and caveats**: Larger values are better, but sometimes the sign is reversed. The log score is ensitive to outliers, as individual negative log score contributions quickly can become very large if the event falls in the tails of the predictive distribution, where $f(y)$ (or p_y) is close to zero. Whether or not that is desirable depends ont the application. In scoringutils, the log score cannot be used for integer-valued forecasts, as the implementation requires a predictive density. In contrast to the crps, the log score is a local scoring rule: it's value only depends only on the probability that was assigned to the actual outcome. This property may be desirable for inferential purposes, for example in a Bayesian context (Winkler et al., 1996). In settings where forecasts inform decision making, it may be more appropriate to score forecasts based on the entire predictive distribution.

Table 4: Detailed explanation of all the metrics, *(continued)*

Metric	Explanation
WIS (Weighted) interval score	The (weighted) interval score is a proper scoring rule for quantile forecasts that converges to the crps for an increasing number of intervals. The score can be decomposed into a sharpness (uncertainty) component and penalties for over- and underprediction. For a single interval, the score is computed as $$IS_\alpha(F, y) = (u - l) + \frac{2}{\alpha} \cdot (l - y) \cdot 1(y \leq l) + \frac{2}{\alpha} \cdot (y - u) \cdot 1(y \geq u),$$ where $1()$ is the indicator function, y is the true value, and l and u are the $\frac{\alpha}{2}$ and $1 - \frac{\alpha}{2}$ quantiles of the predictive distribution F, i.e. the lower and upper bound of a single prediction interval. For a set of K prediction intervals and the median m, the score is computed as a weighted sum, $$WIS = \frac{1}{K + 0.5} \cdot (w_0 \cdot |y - m| + \sum_{k=1}^{K} w_k \cdot IS_\alpha(F, y)),$$ where w_k is a weight for every interval. Usually, $w_k = \frac{\alpha_k}{2}$ and $w_0 = 0.5$. **Usage and caveats**: Smaller scores are better. Applicable to all quantile forecasts, takes the entire predictive distribution into account. Just as the crps, the wis is based on measures of absolute error. When averaging across multiple targets, it will therefore be dominated by targets with higher absolute values. The decomposition into sharpness, over- and underprediction make it easy to interpret scores and use them for model improvement.
DSS Dawid-Sebastiani score	The Dawid-Sebastiani-Score is a proper scoring rule proposed that only relies on the first moments of the predictive distribution and is therefore easy to compute. It is given as $$\text{dss}(F, y) = \left(\frac{y - \mu}{\sigma}\right)^2 + 2 \cdot \log \sigma,$$ where F is the predictive distribution with mean μ and standard deviation σ and y is the true observed value. **Usage and caveats** The dss is applicable to continuous and integer forecasts and easy to compute. Apart from the ease of computation we see little advantage in using it over other scores.

Table 4: Detailed explanation of all the metrics, *(continued)*

Metric	Explanation
Brier score	Proper scoring rule for binary forecasts. The Brier score is computed as $$\text{Brier Score} = \frac{1}{N}\sum_{n=1}^{N}(f_n - y_n),$$ where f_n, with $n = 1, \ldots, N$ are the predicted probablities that the corresponding events, $y_n \in (0, 1)$ will be equal to one.) **Usage**: Applicable to all binary forecasts.
Interval coverage	Interval coverage measures the proportion of observed values that fall in a given prediction interval range. Interval coverage for a single prediction interval range can be calculated as $$IC_\alpha = \text{nominal coverage} - \text{empirical coverage},$$ where nominal coverage is $1 - \alpha$ and empirical coverage is the proportion of true values actually covered by all $1 - \alpha$ prediction intervals. To summarise interval coverage over different over multiple interval ranges, we can compute coverage deviation defined as the mean interval coverage over all K interval ranges α_k with $k = 1, \ldots, K$: $$\text{Coverage deviation} = \frac{1}{K}\sum_{k=1}^{K}\text{IC}_{\alpha_k}$$ **Usage**: Interval coverage for a set of chosen intervals, (e.g. 50% and 90%) gives a good indication of marginal calibration and is easy to interpret. Reporting coverage deviation has the advantage of summarising calibration in a single number, but loses some of the nuance.
Quantile coverage	Quantile coverage for a given quantile level is the proportion of true values smaller than the predictions corresponding to that quantile level. **Usage**: Quantile coverage is similar to interval coverage, but conveys more information. For example, it allows us to look at the 5% and 95% quantile separately, instead of jointly at the 90% prediction interval). This helps to diagnose whether it is the upper or lower end of a prediction interval that is causing problems. Plots of quantile coverage are conceptually very similar to PIT histograms.

Table 4: Detailed explanation of all the metrics, *(continued)*

Metric	Explanation
Probability integral transform (PIT)	The probability integral transform (PIT, Dawid 1984) represents a succinct way to visualise deviations between the predictive distribution F and the true data-generating distribution G. The idea is to transform the observed values such that agreement between forecasts and data can then be examined by observing whether or not the transformed values follow a uniform distribution. The PIT is given by $$u = F(y),$$ where u is the transformed variable and $F(y)$ is the predictive distribution F evaluated at the true observed value y. If $F = G$, then u follows a uniform distribution. For integer outcomes, the PIT is no longer uniform even when forecasts are ideal. Instead, a randomised PIT can be used: $$u = P(y) + v \cdot (P(y) - P(y-1)),$$ where y is again the observed value $P()$ is the cumulative probability assigned to all values smaller or equal to y (where $P(-1) = 0$ by definition, and v is a standard uniform variable independent of y. If P is equal to the true data-generating distribution function, then u is standard uniform. also propose a non-randomised version of the PIT for count data that could be used alternatively. **Usage**: One can plot a histogram of u values to look for deviations from uniformity. U-shaped histograms often result from predictions that are too narrow, while hump-shaped histograms indicate that predictions may be too wide. Biased predictions will usually result in a triangle-shaped histogram. One can also test for deviations from normality, using for example an Anderson-Darling test. This, however, proves to be overly strict in practice and even slight deviations from perfect calibration are punished in a way that makes it very hard to compare models at all. In addition, errors from forecasts may be correlated (i.e. forecasts made on a given date), potentially violating the assumptions of the Anderson-Darling test. We therefore do not recommend it for most use cases.

Table 4: Detailed explanation of all the metrics, *(continued)*

Metric	Explanation
Sharpness	Sharpness is the ability to produce narrow forecasts and is a feature of the forecasts only and does not depend on the observations. Sharpness is therefore only of interest conditional on calibration: a very precise forecast is not useful if it is clearly wrong. As suggested by Funk et al. (2019), we measure sharpness for continuous and integer forecasts represented by predictive samples as the normalised median absolute deviation about the median (MADN)), i.e. $$S(F) = \frac{1}{0.675} \cdot \text{median}(\lvert x - \text{median(x)}\rvert),$$ where x is the vector of all predictive samples and $\frac{1}{0.675}$ is a normalising constant. If the predictive distribution F is the CDF of a normal distribution, then sharpness will equal the standard deviation of F. For quantile forecasts we can directly use the sharpness component of the weighted interval score. Sharpness is then simply the weighted mean of the widths of the central prediction intervals.

Table 4: Detailed explanation of all the metrics, *(continued)*

Metric	Explanation
Bias	Bias is a measure of the tendency of a forecaster to over- or underpredict. For continuous forecasts, bias is given as $$B(F, y) = 1 - 2 \cdot (F(y)),$$ where F is the CDF of the predictive distribution and y is the observed value. For integer-valued forecasts, bias can be calculated as $$B(P, y) = 1 - (P(y) + P(y + 1)),$$ where $P(y)$ is the cumulative probability assigned to all outcomes smaller or equal to y. For quantile forecasts, Bias can be calculated as the maximum percentile rank for which the prediction is smaller than y, if the true value is smaller than the median of the predictive distribution. If the true value is above the median of the predictive distribution, then bias is the minimum percentile rank for which the corresponding quantile is still larger than the true value. If the true value is exactly the median, bias is zero. For a large enough number of quantiles, the percentile rank will equal the proportion of predictive samples below the observed true value, and this metric coincides with the one for continuous forecasts. **Usage**: In contrast to the over- and underprediction penalties of the interval score it is bound between 0 and 1 and represents the tendency of forecasts to be biased rather than the absolute amount of over- and underprediction. It is therefore a more robust measurement, but harder to interpet. It largely depends on the application whether one is more interested in the tendency to be biased or in the absolute value of over- and underpredictions.

Table 4: Detailed explanation of all the metrics, *(continued)*

Metric	Explanation
Mean score ratio	The mean score ratio is used to compare two models on the overlapping set of forecast targets for which both models have made a prediction. The mean score ratio is calculated as the mean score achieved by the first model over the mean score achieved by the second model. More precisely, for two models i, j, we determine the set of overlapping forecasts, denoted by $\mathcal{A}_{ij}$ and compute the mean score ratio θ_{ij} as $$\theta_{ij} = \frac{\text{mean score model } i \text{ on } \mathcal{A}_{ij}}{\text{mean score model } j \text{ on } \mathcal{A}_{ij}}.$$ The mean score ratio can in principle be computed for any arbitrary score. **Usage**: Mean scores ratios are usually calculated in the context of pairwise comparisons, where a set of models is compared by looking at mean score ratios of all possible parings. Whether smaller or larger values are better depends on the orientation of the original score used
Relative skill	Relative skill scores can be used to obtain a ranking of models based on pairwise comparisons between all models. To compute the relative skill θ_i of model i, we take the geometric mean of all mean score ratios that involve model i, i.e. $$\theta_i = \left(\prod_{m=1}^{M} \theta_{im} \right)^{1/M},$$ where M is the number of models. **Usage and caveats**: Relative skill is a helpful way to obtain a model ranking. Whether smaller or larger values are better depends on the orientation of the original score used. It is in principle relatively robust against biases that arise when models only forecast some of the available targets and is a reasonable way to handle missing forecasts. One possible precautionary measure to reduces issues with missing forecasts is to only compare models that have forecasted at least half of all possible targets (this ensures that there is always an overlap between models). If there is no overlap between models, the relative skill implicitly estimates how a model would have forecasted on those missing targets.

References

Angus JE (1994). "The Probability Integral Transform and Related Results." *SIAM Review*, **36**(4), 652–654. ISSN 0036-1445. doi:10.1137/1036146.

Bracher J, Ray EL, Gneiting T, Reich NG (2021a). "Evaluating Epidemic Forecasts in an Interval Format." *PLoS computational biology*, **17**(2), e1008618. ISSN 1553-7358. doi:10.1371/journal.pcbi.1008618.

Bracher J, Wolffram D, Deuschel J, Görgen K, Ketterer JL, Ullrich A, Abbott S, Barbarossa MV, Bertsimas D, Bhatia S, Bodych M, Bosse NI, Burgard JP, Castro L, Fairchild G, Fuhrmann J, Funk S, Gogolewski K, Gu Q, Heyder S, Hotz T, Kheifetz Y, Kirsten H, Krueger T, Krymova E, Li ML, Meinke JH, Michaud IJ, Niedzielewski K, Ożański T, Rakowski F, Scholz M, Soni S, Srivastava A, Zieliński J, Zou D, Gneiting T, Schienle M (2021b). "Short-Term Forecasting of COVID-19 in Germany and Poland during the Second Wave – a Preregistered Study." *medRxiv*, p. 2020.12.24.20248826. doi:10.1101/2020.12.24.20248826.

Bracher J, Wolffram D, Deuschel J, Görgen K, Ketterer JL, Ullrich A, Abbott S, Barbarossa MV, Bertsimas D, Bhatia S, Bodych M, Bosse NI, Burgard JP, Fiedler J, Fuhrmann J, Funk S, Gambin A, Gogolewski K, Heyder S, Hotz T, Kheifetz Y, Kirsten H, Krueger T, Krymova E, Leithäuser N, Li ML, Meinke JH, Miasojedow B, Mohring J, Nouvellet P, Nowosielski JM, Ozanski T, Radwan M, Rakowski F, Scholz M, Soni S, Srivastava A, Gneiting T, Schienle M (2021c). "National and Subnational Short-Term Forecasting of COVID-19 in Germany and Poland, Early 2021." doi:10.1101/2021.11.05.21265810.

Brier GW (1950). "VERIFICATION OF FORECASTS EXPRESSED IN TERMS OF PROBABILITY." *Monthly Weather Review*, **78**(1), 1–3. ISSN 1520-0493, 0027-0644. doi:10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2.

Cramer E, Ray EL, Lopez VK, Bracher J, Brennen A, Rivadeneira AJC, Gerding A, Gneiting T, House KH, Huang Y, Jayawardena D, Kanji AH, Khandelwal A, Le K, Mühlemann A, Niemi J, Shah A, Stark A, Wang Y, Wattanachit N, Zorn MW, Gu Y, Jain S, Bannur N, Deva A, Kulkarni M, Merugu S, Raval A, Shingi S, Tiwari A, White J, Woody S, Dahan M, Fox S, Gaither K, Lachmann M, Meyers LA, Scott JG, Tec M, Srivastava A, George GE, Cegan JC, Dettwiller ID, England WP, Farthing MW, Hunter RH, Lafferty B, Linkov I, Mayo ML, Parno MD, Rowland MA, Trump BD, Corsetti SM, Baer TM, Eisenberg MC, Falb K, Huang Y, Martin ET, McCauley E, Myers RL, Schwarz T, Sheldon D, Gibson GC, Yu R, Gao L, Ma Y, Wu D, Yan X, Jin X, Wang YX, Chen Y, Guo L, Zhao Y, Gu Q, Chen J, Wang L, Xu P, Zhang W, Zou D, Biegel H, Lega J, Snyder TL, Wilson DD, McConnell S, Walraven R, Shi Y, Ban X, Hong QJ, Kong S, Turtle JA, Ben-Nun M, Riley P, Riley S, Koyluoglu U, DesRoches D, Hamory B, Kyriakides C, Leis H, Milliken J, Moloney M, Morgan J, Ozcan G, Schrader C, Shakhnovich E, Siegel D, Spatz R, Stiefeling C, Wilkinson B, Wong A, Gao Z, Bian J, Cao W, Ferres JL, Li C, Liu TY, Xie X, Zhang S, Zheng S, Vespignani A, Chinazzi M, Davis JT, Mu K, y Piontti AP, Xiong X, Zheng A, Baek J, Farias V, Georgescu A, Levi R, Sinha D, Wilde J, Penna ND, Celi LA, Sundar S, Cavany S, España G, Moore S, Oidtman R, Perkins A, Osthus D, Castro L, Fairchild G, Michaud I, Karlen D, Lee EC, Dent J, Grantz KH, Kaminsky J, Kaminsky K, Keegan LT, Lauer

SA, Lemaitre JC, Lessler J, Meredith HR, Perez-Saez J, Shah S, Smith CP, Truelove SA, Wills J, Kinsey M, Obrecht RF, Tallaksen K, Burant JC, Wang L, Gao L, Gu Z, Kim M, Li X, Wang G, Wang Y, Yu S, Reiner RC, Barber R, Gaikedu E, Hay S, Lim S, Murray C, Pigott D, Prakash BA, Adhikari B, Cui J, Rodríguez A, Tabassum A, Xie J, Keskinocak P, Asplund J, Baxter A, Oruc BE, Serban N, Arik SO, Dusenberry M, Epshteyn A, Kanal E, Le LT, Li CL, Pfister T, Sava D, Sinha R, Tsai T, Yoder N, Yoon J, Zhang L, Abbott S, Bosse NI, Funk S, Hellewel J, Meakin SR, Munday JD, Sherratt K, Zhou M, Kalantari R, Yamana TK, Pei S, Shaman J, Ayer T, Adee M, Chhatwal J, Dalgic OO, Ladd MA, Linas BP, Mueller P, Xiao J, Li ML, Bertsimas D, Lami OS, Soni S, Bouardi HT, Wang Y, Wang Q, Xie S, Zeng D, Green A, Bien J, Hu AJ, Jahja M, Narasimhan B, Rajanala S, Rumack A, Simon N, Tibshirani R, Tibshirani R, Ventura V, Wasserman L, O'Dea EB, Drake JM, Pagano R, Walker JW, Slayton RB, Johansson M, Biggerstaff M, Reich NG (2021). "Evaluation of Individual and Ensemble Probabilistic Forecasts of COVID-19 Mortality in the US." *medRxiv*, p. 2021.02.03.21250974. doi:10.1101/2021.02.03.21250974.

Cramer E, Reich NG, Wang SY, Niemi J, Hannan A, House K, Gu Y, Xie S, Horstman S, aniruddhadiga, Walraven R, starkari, Li ML, Gibson G, Castro L, Karlen D, Wattanachit N, jinghuichen, zyt9lsb, aagarwal1996, Woody S, Ray E, Xu FT, Biegel H, GuidoEspana, X X, Bracher J, Lee E, har96, leyouz (2020). "COVID-19 Forecast Hub: 4 December 2020 Snapshot." doi:10.5281/zenodo.3963371.

Dawid AP (1984). "Present Position and Potential Developments: Some Personal Views Statistical Theory the Prequential Approach." *Journal of the Royal Statistical Society: Series A (General)*, **147**(2), 278–290. ISSN 2397-2327. doi:10.2307/2981683.

Dawid AP, Sebastiani P (1999). "Coherent Dispersion Criteria for Optimal Experimental Design." *The Annals of Statistics*, **27**(1), 65–81. ISSN 0090-5364, 2168-8966. doi:10.1214/aos/1018031101.

Elliott G, Timmermann A (2016). "Forecasting in Economics and Finance." *Annual Review of Economics*, **8**(1), 81–110. doi:10.1146/annurev-economics-080315-015346.

Epstein ES (1969). "A Scoring System for Probability Forecasts of Ranked Categories." *Journal of Applied Meteorology*, **8**(6), 985–987. ISSN 0021-8952. doi:10.1175/1520-0450(1969)008<0985:ASSFPF>2.0.CO;2.

European Covid-19 Forecast Hub (2021). "European Covid-19 Forecast Hub." https://covid19forecasthub.eu/.

Funk S, Abbott S, Atkins BD, Baguelin M, Baillie JK, Birrell P, Blake J, Bosse NI, Burton J, Carruthers J, Davies NG, Angelis DD, Dyson L, Edmunds WJ, Eggo RM, Ferguson NM, Gaythorpe K, Gorsich E, Guyver-Fletcher G, Hellewell J, Hill EM, Holmes A, House TA, Jewell C, Jit M, Jombart T, Joshi I, Keeling MJ, Kendall E, Knock ES, Kucharski AJ, Lythgoe KA, Meakin SR, Munday JD, Openshaw PJM, Overton CE, Pagani F, Pearson J, Perez-Guzman PN, Pellis L, Scarabel F, Semple MG, Sherratt K, Tang M, Tildesley MJ, Leeuwen EV, Whittles LK, Group CCW, Team ICCR, Investigators I (2020). "Short-Term Forecasts to Inform the Response to the Covid-19 Epidemic in the UK." *medRxiv*, p. 2020.11.11.20220962. doi:10.1101/2020.11.11.20220962.

Funk S, Camacho A, Kucharski AJ, Lowe R, Eggo RM, Edmunds WJ (2019). "Assessing the Performance of Real-Time Epidemic Forecasts: A Case Study of Ebola in the Western Area Region of Sierra Leone, 2014-15." *PLOS Computational Biology*, **15**(2), e1006785. ISSN 1553-7358. doi:10.1371/journal.pcbi.1006785.

Gelman A, Hwang J, Vehtari A (2014). "Understanding Predictive Information Criteria for Bayesian Models." *Statistics and Computing*, **24**(6), 997–1016. ISSN 1573-1375. doi:10.1007/s11222-013-9416-2.

Gneiting T, Balabdaoui F, Raftery AE (2007). "Probabilistic Forecasts, Calibration and Sharpness." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **69**(2), 243–268. ISSN 1467-9868. doi:10.1111/j.1467-9868.2007.00587.x.

Gneiting T, Raftery AE (2005). "Weather Forecasting with Ensemble Methods." *Science*, **310**(5746), 248–249. ISSN 0036-8075, 1095-9203. doi:10.1126/science.1115255.

Gneiting T, Raftery AE (2007). "Strictly Proper Scoring Rules, Prediction, and Estimation." *Journal of the American Statistical Association*, **102**(477), 359–378. ISSN 0162-1459, 1537-274X. doi:10.1198/016214506000001437.

Good IJ (1952). "Rational Decisions." *Journal of the Royal Statistical Society. Series B (Methodological)*, **14**(1), 107–114. ISSN 0035-9246.

Hamill TM (2001). "Interpretation of Rank Histograms for Verifying Ensemble Forecasts." *Monthly Weather Review*, **129**(3), 550–560. ISSN 1520-0493, 0027-0644. doi:10.1175/1520-0493(2001)129<0550:IORHFV>2.0.CO;2.

Hamner B, Frasco M (2018). *Metrics: Evaluation Metrics for Machine Learning*. R package version 0.1.4, URL https://CRAN.R-project.org/package=Metrics.

Jordan A, Krüger F, Lerch S (2019a). "Evaluating Probabilistic Forecasts with scoringRules." *Journal of Statistical Software*, **90**(12), 1–37. doi:10.18637/jss.v090.i12.

Jordan A, Krüger F, Lerch S (2019b). "Evaluating Probabilistic Forecasts with **scoringRules**." *Journal of Statistical Software*, **90**(12). ISSN 1548-7660. doi:10.18637/jss.v090.i12.

Jose VR (2009). "A Characterization for the Spherical Scoring Rule." *Theory and Decision*, **66**(3), 263–281. ISSN 1573-7187. doi:10.1007/s11238-007-9067-x.

Kukkonen J, Olsson T, Schultz DM, Baklanov A, Klein T, Miranda AI, Monteiro A, Hirtl M, Tarvainen V, Boy M, Peuch VH, Poupkou A, Kioutsioukis I, Finardi S, Sofiev M, Sokhi R, Lehtinen KEJ, Karatzas K, San José R, Astitha M, Kallos G, Schaap M, Reimer E, Jakobs H, Eben K (2012). "A Review of Operational, Regional-Scale, Chemical Weather Forecasting Models in Europe." *Atmospheric Chemistry and Physics*, **12**(1), 1–87. ISSN 1680-7316. doi:10.5194/acp-12-1-2012.

Liboschik T, Fokianos K, Fried R (2017). "tscount: An R Package for Analysis of Count Time Series Following Generalized Linear Models." *Journal of Statistical Software*, **82**(5), 1–51. doi:10.18637/jss.v082.i05.

Machete RL (2012). "Contrasting Probabilistic Scoring Rules." *arXiv:1112.4530 [math, stat]*. 1112.4530.

Mann HB, Whitney DR (1947). "On a Test of Whether One of Two Random Variables Is Stochastically Larger than the Other." *The Annals of Mathematical Statistics*, **18**(1), 50–60. ISSN 0003-4851, 2168-8990. doi:10.1214/aoms/1177730491.

Matheson JE, Winkler RL (1976). "Scoring Rules for Continuous Probability Distributions." *Management Science*, **22**(10), 1087–1096. ISSN 0025-1909. doi:10.1287/mnsc.22.10.1087.

Murphy AH (1971). "A Note on the Ranked Probability Score." *Journal of Applied Meteorology and Climatology*, **10**(1), 155–156. ISSN 1520-0450. doi:10.1175/1520-0450(1971)010<0155:ANOTRP>2.0.CO;2.

R Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Reich NG, Brooks LC, Fox SJ, Kandula S, McGowan CJ, Moore E, Osthus D, Ray EL, Tushar A, Yamana TK, Biggerstaff M, Johansson MA, Rosenfeld R, Shaman J (2019). "A Collaborative Multiyear, Multimodel Assessment of Seasonal Influenza Forecasting in the United States." *Proceedings of the National Academy of Sciences*, **116**(8), 3146–3154. ISSN 0027-8424, 1091-6490. doi:10.1073/pnas.1812594116.

Timmermann A (2018). "Forecasting Methods in Finance." *Annual Review of Financial Economics*, **10**(1), 449–479. doi:10.1146/annurev-financial-110217-022713.

Winkler RL, Muñoz J, Cervera JL, Bernardo JM, Blattenberger G, Kadane JB, Lindley DV, Murphy AH, Oliver RM, Ríos-Insua D (1996). "Scoring Rules and the Evaluation of Probabilities." *Test*, **5**(1), 1–60. ISSN 1863-8260. doi:10.1007/BF02562681.

Yan Y (2016). *MLmetrics: Machine Learning Evaluation Metrics*. R package version 1.1.1, URL https://CRAN.R-project.org/package=MLmetrics.

Zeileis A, Lang MN (2022). *topmodels: Infrastructure for Inference and Forecasting in Probabilistic Models*. R package version 0.1-0/r1498, URL https://R-Forge.R-project.org/projects/topmodels/.

Affiliation:

Nikos I. Bosse
London School of Hygiene & Tropical Medicine (LSHTM)
Centre for Mathematical Modelling of Infectious Diseases
London School of Hygiene & Tropical Medicine
Keppel Street
London WC1E 7HT
E-mail: nikos.bosse@lshtm.ac.uk
URL: https://lshtm.ac.uk

Hugo Gruson
LSHTM
Centre for Mathematical Modelling of Infectious Diseases
London School of Hygiene & Tropical Medicine
Keppel Street
London WC1E 7HT
E-mail: hugo.gruson@lshtm.ac.uk

Anne Cori
Imperial College London
MRC Centre for Global Infectious Disease Analysis, School of Public Health
Imperial College London
Norfolk Place
London W2 1PG
E-mail: a.cori@imperial.ac.uk

Edwin van Leeuwen
UK Health Security Agency, LSHTM
Statistics, Modelling and Economics Department
UK Health Security Agency
London NW9 5EQ
E-mail: Edwin.VanLeeuwen@phe.gov.uk

Sebastian Funk
LSHTM
Centre for Mathematical Modelling of Infectious Diseases
London School of Hygiene & Tropical Medicine
Keppel Street
London WC1E 7HT
E-mail: sebastian.funk@lshtm.ac.uk

Sam Abbott
LSHTM
Centre for Mathematical Modelling of Infectious Diseases
London School of Hygiene & Tropical Medicine
Keppel Street
London WC1E 7HT
E-mail: sam.abbott@lshtm.ac.uk

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd