




calzone: A Python package for measuring calibration of probabilistic models for classification

Kwok Lung Fan¹, Gene Pennello¹, Qi Liu¹, Nicholas Petrick¹, Ravi K. Samala¹, Frank W. Samuelson¹, Yee Lam Elim Thompson¹, and Qian Cao^{1¶}

¹ U.S. Food and Drug Administration ¶ Corresponding author

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Open Journals](#) 

Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#))

Summary

calzone is a Python package for evaluating the calibration of probabilistic outputs of classifier models. It provides a set of functions for visualizing calibration and computation of calibration metrics given a representative dataset with the model's predictions and true class labels. The metrics provided in calzone include: Expected Calibration Error (ECE), Maximum Calibration Error (MCE), Hosmer-Lemeshow (HL) statistic, Integrated Calibration Index (ICI), Spiegelhalter's Z-statistics and Cox's calibration slope/intercept. The package is designed with versatility in mind. For many of the metrics, users can adjust the binning scheme and toggle between top-class or class-wise calculations.

Statement of need

Classification is one of the most common applications in machine learning. Classification models are often evaluated by a proper scoring rule - a scoring function that assigns the best score when predicted probabilities match the true probabilities - such as cross-entropy or mean square error ([Gneiting & Raftery, 2007](#)). Examination of the discrimination performance (resolution), such as AUC or Se/Sp are also used to evaluate model performance. However, the reliability or calibration performance of the model is often overlooked.

Diamond ([1992](#)) showed that the resolution (i.e., high performance) of a model does not indicate the reliability/calibration (i.e., how well predicted probabilities match true probabilities) of the model. Bröcker ([2009](#)) later showed that any proper scoring rule can be decomposed into the resolution and reliability. Thus, even if the model has high resolution, it may not be a reliable model. In many high-risk machine learning applications, such as medical diagnosis, the reliability of the model could be important as it could impact the clinician's interpretation.

We define calibration as the agreement between the predicted probability and the true posterior probability of a class-of-interest, $P(D = 1|\hat{p} = p) = p$. This has been defined as moderate calibration by Van Calster & Steyerberg ([2018](#)) .

In the calzone package, we provide a set of functions and classes for visualizing calibration and evaluating calibration metrics given a representative dataset from the intended population. Existing libraries such as scikit-learn lacks calibration metrics that are widely used in the statistical literature. Other libraries such as uncertainty-toolbox are focused on implementing calibration methods but do not include any calibration assessment. ([Chung et al., 2021](#)).

37 Functionality

38 Reliability Diagram

39 The reliability diagram (also referred to as the calibration plot) is a graphical representation of
40 the calibration of a classification model (Bröcker & Smith, 2007; Murphy & Winkler, 1977). It
41 groups the predicted probabilities into bins and plots the mean predicted probability against the
42 empirical frequency in each bin. The reliability diagram can be used to assess the calibration
43 of the model and to identify any systematic errors in the predictions. In addition, calzone
44 gives the option to also plot the confidence interval of the empirical frequency in each bin.
45 The confidence intervals are calculated using the Wilson's score interval (Wilson, 1927). We
46 provide example data in the example_data folder which are simulated using a beta-binomial
47 distribution (Griffiths, 1973). The predicted probabilities are sampled from a beta distribution
48 and the true labels are assigned by performing Bernoulli trials with the sampled probabilities.
49 Users can generate simulated data using the fake_binary_data_generator class in the utils
50 module. Figure Figure 1 shows an example of the reliability diagram with 15 equal-width bins
51 for a well-calibrated dataset, where the x-axis is the mean predicted probability and the y-axis
52 is the empirical frequency.

```
from calzone.utils import reliability_diagram
from calzone.vis import plot_reliability_diagram

wellcal_dataloader = data_loader(
    data_path="example_data/simulated_welldata.csv"
)

reliability, confidence, bin_edges, bin_counts = reliability_diagram(
    wellcal_dataloader.labels,
    wellcal_dataloader.probs,
    num_bins=15,
    class_to_plot=1
)

plot_reliability_diagram(
    reliability,
    confidence,
    bin_counts,
    error_bar=True,
    title='Class 1 reliability diagram for well calibrated data'
)
```

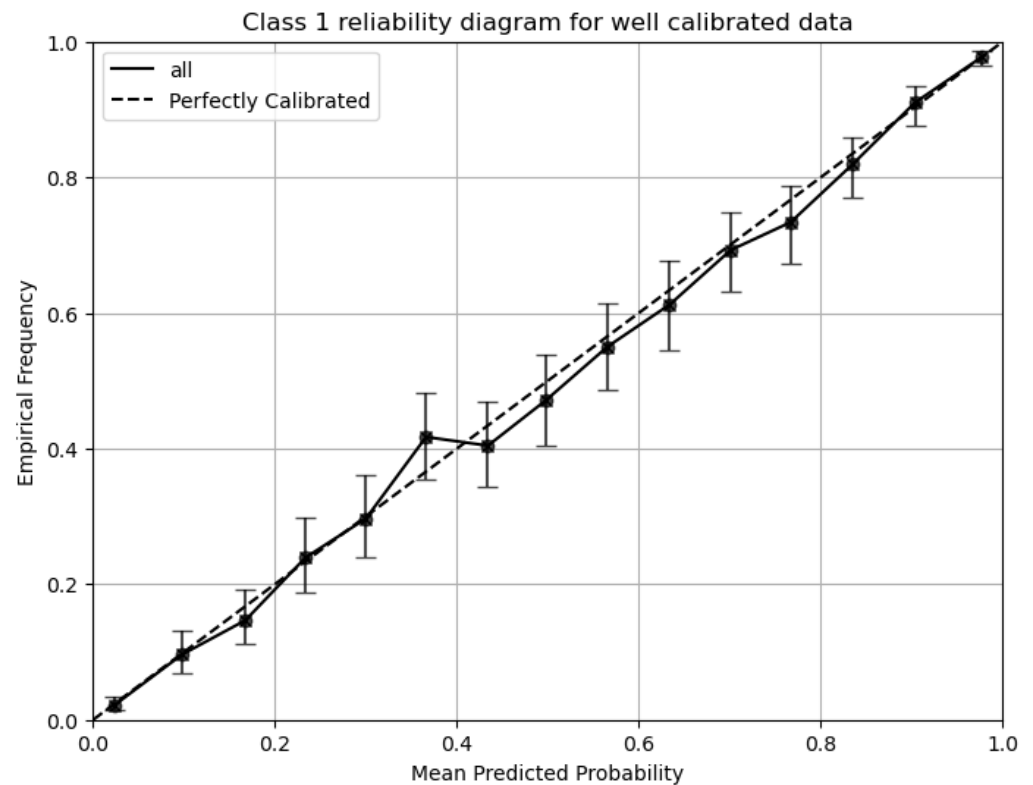


Figure 1: Reliability Diagram for well calibrated data.

Calibration metrics

calzone provides functions to compute various calibration metrics. The `CalibrationMetrics()` class allows the user to compute the calibration metrics in a more convenient way. The following are metrics that are currently supported in calzone:

Expected Calibration Error (ECE) and Maximum Calibration Error (MCE)

Expected Calibration Error (ECE) and Maximum Calibration Error (MCE) (Guo et al., 2017; Pakdaman Naeini et al., 2015) aim to measure the average and maximum absolute deviation between predicted probability and true probability. We provide the option to use equal-width binning or equal-count binning, labeled as ECE-H and ECE-C respectively. Users can also choose to compute the metrics for the class-of-interest or the top-class. Top-class mean only the predicted class calibration will be evaluated. In the case of class-of-interest, calzone will evaluate the calibration of a one-vs-rest classification problem. The following snippet demonstrates how these metrics are calculated in our package:

```
from calzone.metrics import calculate_ece_mce

reliability, confidence, bin_edges, bin_counts = reliability_diagram(
    wellcal_dataloader.labels,
    wellcal_dataloader.probs,
    num_bins=10,
    class_to_plot=1,
    is_equal_freq=False
)

### Both ECE and MCE are calculated at the same time
```

```
ece_h_classone, mce_h_classone = calculate_ece_mce(
    reliability,
    confidence,
    bin_counts=bin_counts
)
```

66 Hosmer-Lemeshow statistic (HL)

67 The Hosmer-Lemeshow (HL) statistical test is for evaluating the calibration of a probabilistic
68 model. It is a chi-square-based test that compares the observed and expected number of
69 events in each bin. The null hypothesis is that the model is well calibrated. HL-test first bins
70 data into predicted probability bins (equal-width H or equal-count C) and the test statistic is
71 calculated as:

$$HL = \sum_{m=1}^M \frac{(O_{1,m} - E_{1,m})^2}{E_{1,m}(1 - \frac{E_{1,m}}{N_m})} \sim \chi_{M-2}^2$$

72 where $E_{1,m}$ is the expected number of class-of-interest events in the m^{th} bin, $O_{1,m}$ is the
73 observed number of class-of-interest events in the m^{th} bin, N_m is the total number of
74 observations in the m^{th} bin, and M is the number of bins. In calzone, the HL-test can be
75 computed as follows:

```
from calzone.metrics import hosmer_lemeshow_test

HL_H_ts, HL_H_p, df = hosmer_lemeshow_test(
    reliability,
    confidence,
    bin_count=bin_counts
)
```

76 When performing the HL test on validation sets that are not used in training, the degree
77 of freedom of the HL test changes from $M - 2$ to M (Hosmer Jr et al., 2013). Intuitively,
78 $\frac{(O_{1,m} - E_{1,m})^2}{E_{1,m}(1 - \frac{E_{1,m}}{N_m})}$ is the difference squared divided by the variance of a binomial distribution and
79 follows a chi-square distribution with 1 degree of freedom. Hence, the sum of M chi-square
80 distributions with 1 degree of freedom is a chi-square distribution with M degrees of freedom if
81 the data has no effect on the model. The increase in degree of freedom for validation samples
82 has often been overlooked but it is crucial for the test to maintain the correct type 1 error
83 rate. In calzone, the default degree of freedoms is $M - 2$ and users can specify the degree of
84 freedom of the HL test by setting the df parameter.

85 Cox's calibration slope/intercept

86 Cox's calibration slope/intercept is a regression analysis method for assessing the calibration
87 of a probabilistic model (COX, 1958), which doesn't require binning. A logistic regression
88 model is fit to the data, with the predicted odds ($\frac{p}{1-p}$) as the independent variable and the
89 outcome as the dependent variable. The slope and intercept of the regression line are then
90 used to assess the calibration of the model. A slope of 1 and intercept of 0 indicates perfect
91 calibration. To test whether the model is calibrated, fix the slope to 1 and fit the intercept.
92 If the intercept is significantly different from 0, the model is not calibrated. Then, fix the
93 intercept to 0 and fit the slope. If the slope is significantly different from 1, the model is not
94 calibrated. Alternatively, the slope and intercept can be fitted and tested simultaneously using
95 a bivariate distribution (McCullagh & Nelder, 1989). This feature is not provided in calzone
96 but user can extract the covariance matrix by printing the result and perform the test manually.
97 In calzone, Cox's calibration slope/intercept can be computed as follows:

```
from calzone.metrics import cox_regression_analysis
```

```
cox_slope, cox_intercept, cox_slope_ci, cox_intercept_ci = cox_regression_analysis(  
    wellcal_dataloader.labels,  
    wellcal_dataloader.probs,  
    class_to_calculate=1,  
    print_results=True,  
    fix_slope=True  
)
```

98 The slope and intercept values indicate the type of miscalibration. A slope >1 shows overcon-
99 fidence at high probabilities and underconfidence at low probabilities (and vice versa). In other
100 word, a slope < 1 (> 1) indicates that the spread of the predicted risks is too large (small)
101 relative to the true risks. A positive intercept indicates general overconfidence (and vice versa).
102 However, even with ideal slope and intercept values, the model may still be miscalibrated due
103 to non-linear effects that Cox's analysis cannot detect.

104 Integrated calibration index (ICI)

105 The integrated calibration index (ICI) is very similar to Expected calibration error (ECE). It
106 also tries to measure the average deviation between predicted probability and true probability.
107 However, ICI does not use binning to estimate the true probability of a group of samples with
108 similar predicted probability. Instead, ICI uses curve smoothing techniques to fit the regression
109 curve and uses the regression result as the true probability (Austin & Steyerberg, 2019). The
110 ICI is then calculated using the following formula:

$$ICI = \frac{1}{n} \sum_{i=1}^n |f(p_i) - p_i|$$

111 where f is the fitting function and p is the predicted probability. The curve fitting is usually
112 done with Locally Weighted Scatterplot Smoothing (LOWESS). However, it is possible to use
113 any curve fitting method to calculate the ICI. One possible alternative is to use the Cox's
114 calibration result and calculate the average difference between the predicted probability and
115 the estimated true probability from the curve. In calzone, we provide Cox's ICI and loess
116 ICI support while the user can also use any curve fitting method to calculate the ICI using
117 functions in calzone.

```
from calzone.metrics import (  
    cox_regression_analysis,  
    lowess_regression_analysis,  
    cal_ICI_cox  
)
```

```
### calculating cox ICI
```

```
cox_ici = cal_ICI_cox(  
    cox_slope,  
    cox_intercept,  
    wellcal_dataloader.probs,  
    class_to_calculate=1  
)
```

```
### calculating LOWESS ICI
```

```
loess_ici, lowess_fit_p, lowess_fit_p_correct = lowess_regression_analysis(  
    wellcal_dataloader.labels,  
    wellcal_dataloader.probs,  
    class_to_calculate=1,  
    span=0.5,  
    delta=0.001,
```

```
        it=0
    )
```

118 Notice that flexible curve fitting methods such as LOWESS regression are very sensitive to
119 the choice of span and delta parameters. The user can visualize the fitting result to avoid
120 overfitting or underfitting.

121 Spiegelhalter's Z-test

122 Spiegelhalter's Z-test is a test of calibration proposed by Spiegelhalter in 1986 (Spiegelhalter,
123 1986). It uses the fact that the Brier score can be decomposed into:

$$B = \frac{1}{N} \sum_{i=1}^N (x_i - p_i)^2 = \frac{1}{N} \sum_{i=1}^N (x_i - p_i)(1 - 2p_i) + \frac{1}{N} \sum_{i=1}^N p_i(1 - p_i)$$

124 And the test statistic (TS) of Z test is defined as:

$$Z = \frac{B - E(B)}{\sqrt{\text{Var}(B)}} = \frac{\sum_{i=1}^N (x_i - p_i)(1 - 2p_i)}{\sum_{i=1}^N (1 - 2p_i)^2 p_i(1 - p_i)}$$

125 and it is asymptotically distributed as a standard normal distribution. In calzone, it can be
126 calculated using:

```
from calzone.metrics import spiegelhalter_z_test
```

```
z, p_value = spiegelhalter_z_test(
    wellcal_dataloader.labels,
    wellcal_dataloader.probs,
    class_to_calculate=1
)
```

127 Metrics class

128 calzone also provides a class called CalibrationMetrics() to calculate all the metrics men-
129 tioned above. The user can also use this class to calculate a list of metrics or all the metrics
130 within a single function call.

```
from calzone.metrics import CalibrationMetrics
```

```
metrics = CalibrationMetrics(class_to_calculate=1)
```

```
CalibrationMetrics.calculate_metrics(
    wellcal_dataloader.labels,
    wellcal_dataloader.probs,
    metrics='all'
)
```

131 Other features

132 Confidence intervals

133 In addition to point estimates of calibration performance, calzone also provides functionality
134 to compute confidence intervals for all metrics. For most metrics, this is computed through
135 bootstrapping. The only exception is the confidence intervals from the reliability diagram. The
136 user can specify the number of bootstrap samples and the confidence level.

```
from calzone.metrics import CalibrationMetrics
```

```
metrics = CalibrationMetrics(class_to_calculate=1)
```

```
CalibrationMetrics.bootstrap(
    wellcal_data_loader.labels,
    wellcal_data_loader.probs,
    metrics='all',
    n_samples=1000
)
```

and a structured NumPy array will be returned.

Subgroup analysis

calzone will perform subgroup analysis by default in the command line user interface. If the user input CSV file contains a subgroup column, the program will compute metrics for the entire dataset and for each subgroup. A detailed description of the input format can be found in the documentation.

Prevalence adjustment

calzone also provides prevalence adjustment to account for prevalence changes between training data and testing data. Since calibration is defined using posterior probability, a mere shift in the disease prevalence of the testing data will result in miscalibration. It can be fixed by searching for the optimal derived original prevalence such that the adjusted probability minimizes a proper scoring rule such as cross-entropy loss. The formula of prevalence adjusted probability is:

$$P'(D = 1 | \hat{p} = p) = \frac{\eta' / (1 - \eta')}{(1/p - 1)(\eta / (1 - \eta))} = p'$$

where η is the prevalence of the testing data, η' is the prevalence of the training data, and p is the predicted probability (Chen et al., 2018; Gu & Pepe, 2010; Horsch et al., 2008; Tian et al., 2020). We search for the optimal η' that minimizes the cross-entropy loss. The user can also specify η' and adjust the probability output directly if the training set prevalence is available.

Multiclass extension

calzone also provides a multiclass extension to calculate the metrics for multiclass classification. The user can specify the class to calculate the metrics using a 1-vs-rest approach and test the calibration of each class. Alternatively, the user can transform the data and make the problem become a top-class calibration problem. The top-class calibration has a similar format to binary classification, but the class 0 probability is defined as 1 minus the probability of the class with the highest probability, and the class 1 probability is defined as the probability of the class with the highest probability. The labels are transformed into whether the predicted class equals the true class, 0 if not and 1 if yes. Notice that the interpretation of some metrics may change in the top-class transformation.

Verification of methods

We compared the results calculated by calzone with external packages for some metrics to ensure the correctness of the implementation. For the reliability diagram verification, we compared the result with the `sklearn.calibration.calibration_curve()` function in scikit-learn (Pedregosa et al., 2011). For the top-class ECE and Spiegelhalter's Z scores, we compared the result with the MAPIE package (Taquet et al., 2022). For the Hosmer-Lemeshow statistic, we compared the result with the ResourceSelection package in R language (Lele et

171 al., 2024). Their results are consistent with ours. For other metrics such as ICI, no external
172 package is available, so we compared the result with ECE as they both measure the average
173 absolute difference. We obtained reasonably similar results. We include the verification codes
174 and comparison in our documentation.

175 Command line interface

176 calzone also provides a command line interface. Users can visualize the calibration curve,
177 calculate calibration metrics and their confidence intervals using the this. For help on running
178 this functionality, the user can run `python cal_metrics.py -h`.

179 Acknowledgements

180 The mention of commercial products, their sources, or their use in connection with material
181 reported herein is not to be construed as either an actual or implied endorsement of such
182 products by the Department of Health and Human Services. This is a contribution of the U.S.
183 Food and Drug Administration and is not subject to copyright.

184 The authors acknowledge the Research Participation Program at the Center for Devices and
185 Radiological Health administered by the Oak Ridge Institute for Science and Education through
186 an interagency agreement between the U.S. Department of Energy and the U.S. Food and
187 Drug Administration.

188 Conflicts of interest

189 The authors declare no conflicts of interest.

190 References

- 191 Austin, P. C., & Steyerberg, E. W. (2019). The integrated calibration index (ICI) and related
192 metrics for quantifying the calibration of logistic regression models. *Statistics in Medicine*,
193 38(21), 4051–4065. <https://doi.org/https://doi.org/10.1002/sim.8281>
- 194 Bröcker, J. (2009). Reliability, sufficiency, and the decomposition of proper scores. *Quarterly*
195 *Journal of the Royal Meteorological Society*, 135(643), 1512–1519. <https://doi.org/https://doi.org/10.1002/qj.456>
- 196 Bröcker, J., & Smith, L. A. (2007). Increasing the reliability of reliability diagrams. *Weather*
197 *and Forecasting*, 22(3), 651–661. <https://doi.org/10.1175/WAF993.1>
- 198 Chen, W., Sahiner, B., Samuelson, F., Pezeshk, A., & Petrick, N. (2018). Calibration of
199 medical diagnostic classifier scores to the probability of disease. *Statistical Methods in*
200 *Medical Research*, 27(5), 1394–1409. <https://doi.org/10.1177/0962280216661371>
- 201 Chung, Y., Char, I., Guo, H., Schneider, J., & Neiswanger, W. (2021). Uncertainty toolbox:
202 An open-source library for assessing, visualizing, and improving uncertainty quantification.
203 *arXiv Preprint arXiv:2109.10254*.
- 204 COX, D. R. (1958). Two further applications of a model for binary regression. *Biometrika*,
205 45(3-4), 562–565. <https://doi.org/10.1093/biomet/45.3-4.562>
- 206 Diamond, G. A. (1992). What price perfection? Calibration and discrimination of clinical
207 prediction models. *Journal of Clinical Epidemiology*, 45(1), 85–89. [https://doi.org/https://doi.org/10.1016/0895-4356\(92\)90192-P](https://doi.org/https://doi.org/10.1016/0895-4356(92)90192-P)
- 208 Gneiting, T., & Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation.
209 *Journal of the American Statistical Association*, 102(477), 359–378.
- 210
- 211

- 212 Griffiths, D. A. (1973). Maximum likelihood estimation for the beta-binomial distribution
213 and an application to the household distribution of the total number of cases of a disease.
214 *Biometrics*, 29(4), 637–648. <http://www.jstor.org/stable/2529131>
- 215 Gu, W., & Pepe, M. S. (2010). Estimating the diagnostic likelihood ratio of a continuous
216 marker. *Biostatistics*, 12(1), 87–101. <https://doi.org/10.1093/biostatistics/kxq045>
- 217 Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural
218 networks. In D. Precup & Y. W. Teh (Eds.), *Proceedings of the 34th international
219 conference on machine learning* (Vol. 70, pp. 1321–1330). PMLR. [https://proceedings.
220 mlr.press/v70/guo17a.html](https://proceedings.mlr.press/v70/guo17a.html)
- 221 Horsch, K., Giger, M. L., & Metz, C. E. (2008). Prevalence scaling: Applications to an
222 intelligent workstation for the diagnosis of breast cancer. *Academic Radiology*, 15(11),
223 1446–1457. <https://doi.org/https://doi.org/10.1016/j.acra.2008.04.022>
- 224 Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied logistic regression*.
225 John Wiley & Sons.
- 226 Lele, S. R., Keim, J. L., & Solymos, P. (2024). *ResourceSelection: Resource selection (proba-
227 bility) functions for use-availability data*. <https://github.com/psolymos/ResourceSelection>
- 228 McCullagh, P., & Nelder, J. A. (1989). *Generalized linear models*. Chapman & Hall / CRC.
- 229 Murphy, A. H., & Winkler, R. L. (1977). Reliability of subjective probability forecasts of
230 precipitation and temperature. *Journal of the Royal Statistical Society. Series C (Applied
231 Statistics)*, 26(1), 41–47. <http://www.jstor.org/stable/2346866>
- 232 Pakdaman Naeini, M., Cooper, G., & Hauskrecht, M. (2015). Obtaining well calibrated
233 probabilities using bayesian binning. *Proceedings of the AAAI Conference on Artificial
234 Intelligence*, 29(1). <https://doi.org/10.1609/aaai.v29i1.9602>
- 235 Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M.,
236 Prettenhofer, P., Weiss, R., Dubourg, V., & others. (2011). Scikit-learn: Machine learning
237 in python. *Journal of Machine Learning Research*, 12(Oct), 2825–2830.
- 238 Spiegelhalter, D. J. (1986). Probabilistic prediction in patient management and clinical trials.
239 *Statistics in Medicine*, 5(5), 421–433.
- 240 Taquet, V., Blot, V., Morzadec, T., Lacombe, L., & Brunel, N. (2022). MAPIE: An open-source
241 library for distribution-free uncertainty quantification. *arXiv Preprint arXiv:2207.12274*.
- 242 Tian, J., Liu, Y.-C., Glaser, N., Hsu, Y.-C., & Kira, Z. (2020). Posterior re-calibration
243 for imbalanced datasets. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan,
244 & H. Lin (Eds.), *Advances in neural information processing systems* (Vol. 33, pp.
245 8101–8113). Curran Associates, Inc. [https://proceedings.neurips.cc/paper_files/paper/
246 2020/file/5ca359ab1e9e3b9c478459944a2d9ca5-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2020/file/5ca359ab1e9e3b9c478459944a2d9ca5-Paper.pdf)
- 247 Van Calster, B., & Steyerberg, E. W. (2018). Calibration of prognostic risk scores. In
248 *Wiley StatsRef: Statistics reference online* (pp. 1–10). John Wiley & Sons, Ltd.
249 <https://doi.org/https://doi.org/10.1002/9781118445112.stat08078>
- 250 Wilson, E. B. (1927). Probable inference, the law of succession, and statistical inference.
251 *Journal of the American Statistical Association*, 22(158), 209–212.