

Stacked conformal prediction

Paulo C. Marques F.

PAULOCMF1@INSPER.EDU.BR

Inspier Institute of Education and Research, Rua Quatá 300, São Paulo 04546-042, Brazil

Editor: Khuong An Nguyen, Zhiyuan Luo, Harris Papadopoulos, Tuwe Löfström, Lars Carlsson and Henrik Boström

Abstract

We consider a method for conformalizing a stacked ensemble of predictive models, showing that the potentially simple form of the meta-learner at the top of the stack enables a procedure with manageable computational cost that achieves approximate marginal validity without requiring the use of a separate calibration sample. Empirical results indicate that the method compares favorably to a standard inductive alternative.

Keywords: Supervised learning, prediction sets, conformal prediction, stacking, exchangeability.

1. Introduction

In its most general transductive (full) form, conformal prediction (Vovk et al., 1999, 2005) creates a tension between desirable statistical properties and computational complexity: while it always guarantees, for exchangeable data, the finite-sample marginal coverage of the nonparametric, model-agnostic prediction sets produced by the method, it may, in certain settings, require a highly intensive computational process, due to the necessity of repeatedly retraining the underlying predictive model. Inductive (split) conformal prediction (Papadopoulos et al., 2002; Lei et al., 2018) is an appealing and widely adopted variation that balances the computational cost through a trade-off: while preserving the statistical properties of the full conformal prediction procedure and requiring a single training process for the specified predictive model, it forces the available data to be split into training and calibration samples, which may result in neither building the most accurate predictive model, nor obtaining the most efficient confidence assessment of the model’s generalization capacity. Several alternatives have been developed to keep the computational cost under control while making a more symmetric use of the available data, starting with cross-conformal prediction (Vovk, 2015), which inspired the development of the jackknife+ (Barber et al., 2021) and the nested conformal prediction framework (Gupta et al., 2022).

In this work, we develop a method for conformalizing a stacked ensemble of predictive models (Wolpert, 1992; Breiman, 1996). This construction serves a dual purpose. First, it leverages the usual predictive gains of model stacking. Second, it takes advantage of the fact that the meta-learner at the top of the stack can be a simpler model, enabling a low-cost conformalization procedure that eliminates the need for a separate calibration set while producing prediction intervals with approximate marginal validity.

We proceed in the following manner. In Section 2, in a regression setting with exchangeable data, we construct a stack of predictive models that becomes totally symmetric by including the future observable pair in the training sample used to fit the base-learners at the first stack level. For this symmetric stack, we prove that the assumed exchangeability

of the training sample is transferred to the second level of the stack. Consequently, for an arbitrary meta-learner placed at the top of the stack, we have the exact marginal validity of a random conformal prediction set associated with the future observable pair. Section 3 makes the stack construction feasible by moving the future observable pair out of the training sample. This breaks the distributional symmetry of the stack, but, under a stability assumption for the base stack models, the previously stated marginal validity property still holds approximately. In Section 4, we specialize the meta-learner to multiple linear regression to take advantage of the fact that for this simple model a full conformalization procedure can be carried out at low computational cost through classical rank-one matrix updates. We apply the resulting stacked conformal prediction procedure to two datasets, comparing the stacked conformal prediction intervals to those produced by the inductive conformalized quantile regression method. In Section 5, we present our concluding remarks and give pointers to open source software implementations of the developed method.

2. Symmetric stack

In a supervised learning setting, we have random pairs $\{(X_i, Y_i)\}_{i=1}^{n+1}$, for which the features vector $X_i \in \mathbb{R}^d$ and the response variable $Y_i \in \mathbb{R}$. In this regression problem, we refer to (X_{n+1}, Y_{n+1}) as the future observable pair. Let \mathbb{S}_n denote the set of $n \times n$ permutation matrices, which are square matrices with exactly one entry equal to 1 in each row and column, and all other entries equal to zero. Laying out our training sample — which for now includes the future observable pair — as the $(n+1) \times (d+1)$ random matrix

$$T = \begin{pmatrix} X_{1,1} & X_{1,2} & \cdots & X_{1,d} & Y_1 \\ X_{2,1} & X_{2,2} & \cdots & X_{2,d} & Y_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ X_{n,1} & X_{n,2} & \cdots & X_{n,d} & Y_n \\ X_{n+1,1} & X_{n+1,2} & \cdots & X_{n+1,d} & Y_{n+1} \end{pmatrix},$$

we make the assumption that the data are exchangeable, meaning that T and ΠT have the same distribution, for every permutation matrix $\Pi \in \mathbb{S}_{n+1}$.

We randomly divide the training sample into $K \geq 2$ folds of size $t = (n+1)/K$, assuming that $n+1$ is divisible by K , by means of an $(n+1) \times (n+1)$ random permutation matrix Q , which we refer to as the folding scheme matrix. We suppose that Q is uniformly distributed, so that $\Pr(Q = q) = 1/(n+1)!$, for every $q \in \mathbb{S}_{n+1}$, and we assume that this folding scheme matrix Q and the training data T are independent. We use the block notation

$$Q = \begin{pmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_K \end{pmatrix},$$

in which the realizations of the matrix blocks Q_k are in $\mathbb{R}^{t \times (n+1)}$, for $k = 1, \dots, K$. For the folding scheme defined by the matrix Q , let ϕ be a fold indicator function, defined by $\phi(i) = \sum_{k=1}^K k \cdot I(\sum_{\ell=1}^t (Q_k)_{\ell,i} = 1)$, meaning that $\phi(i)$ determines the fold number k to which the i -th training sample unit has been assigned, for $i = 1, \dots, n+1$. Moreover, using

the notation introduced for the Q blocks, we define, for $k = 1, \dots, K$, the k -th fold exclusion matrix

$$Q_{\setminus k} = \begin{pmatrix} Q_1 \\ \vdots \\ Q_{k-1} \\ Q_{k+1} \\ \vdots \\ Q_K \end{pmatrix},$$

whose realizations are in $\mathbb{R}^{(n+1-t) \times (n+1)}$.

We define a symmetric stacked ensemble of predictive models as follows. The stack is built from $M \geq 1$ base learning methods. For each training sample unit, the goal is to use its features vector to predict the corresponding response variable using models trained only on training data folds that do not include the training sample unit under consideration. Formally, for $m = 1, \dots, M$, we have prediction functions

$$\hat{\mu}_m : \mathbb{R}^{(n+1-t) \times (d+1)} \times \mathbb{R}^d \rightarrow \mathbb{R},$$

and we assume that each learning method treats its training data $S \in \mathbb{R}^{(n+1-t) \times (d+1)}$ symmetrically, so that

$$\hat{\mu}_m(S, x) = \hat{\mu}_m(\Pi S, x), \quad (\star)$$

for every permutation matrix $\Pi \in \mathbb{S}_{n+1-t}$ and each $x \in \mathbb{R}^d$. When the learning method involves some form of randomization — such as the bootstrap process in Random Forests (Breiman, 2001), or the stochastic gradient descent optimization in Deep Neural Networks (Bishop and Bishop, 2023) — we assume, without loss of generality, that the seed of the underlying pseudo-random number generator is set using a symmetric hash function of the training data S . The stack base-learners make predictions

$$Z_i = (Z_{i,1}, \dots, Z_{i,M}) \in \mathbb{R}^M,$$

in which $Z_{i,m} = \hat{\mu}_m(Q_{\setminus \phi(i)} T, X_i)$, for $i = 1, \dots, n+1$. This totally symmetric definition of the stack implies that the assumed exchangeability of the training sample T is transferred to the next stack level, as stated by the following result, proved in the Appendix.

Proposition 1 *The second-level random pairs $(Z_1, Y_1), \dots, (Z_{n+1}, Y_{n+1})$ are exchangeable for a symmetric stack.*

The following result, proved in the Appendix, formalizes the idea that, for a generic meta-learner placed at the top of the symmetric stack, Proposition 1 and a standard conformal argument allow us to construct a random prediction set for the future observable pair with exact marginal validity. We denote the ceiling of $u \in \mathbb{R}$ by $\lceil u \rceil = \min\{z \in \mathbb{Z} : u \leq z\}$.

Proposition 2 *Let $\hat{\psi}_n^{(y)}$ be a meta-learner trained from $\{(Z_1, Y_1), \dots, (Z_n, Y_n), (Z_{n+1}, y)\}$, for $y \in \mathbb{R}$, and suppose the order of the $n+1$ pairs in this sample is irrelevant for the construction of $\hat{\psi}_n^{(y)}$. For a conformity function $\rho : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$, define the conformity scores $R_i^{(y)} = \rho(Y_i, \hat{\psi}_n^{(y)}(Z_i))$, for $i = 1, \dots, n+1$, and let $R_{(1)}^{(y)} \leq R_{(2)}^{(y)} \leq \dots \leq R_{(n)}^{(y)}$ denote the*

ordered conformity scores among $\{R_1^{(y)}, R_2^{(y)}, \dots, R_n^{(y)}\}$. Choosing a nominal miscoverage level $0 < \alpha < 1$ such that $\lceil (1 - \alpha)(n + 1) \rceil \leq n$, we have that

$$\Pr(Y_{n+1} \in C_{n+1}^{(\alpha)}) \geq 1 - \alpha,$$

in which we defined the random prediction set $C_{n+1}^{(\alpha)} = \left\{ y \in \mathbb{R} : R_{n+1}^{(y)} \leq R_{(\lceil (1-\alpha)(n+1) \rceil)}^{(y)} \right\}$.

3. Feasible stack

The symmetric stack described in Section 2 is an oracle construct that cannot be implemented in practice, since at training time we do not know the observed value of the future response Y_{n+1} . A *feasible stack* is attained by removing the future observable pair (X_{n+1}, Y_{n+1}) from the training sample. In doing so, we break the distributional symmetry of the stack, but if the predictions made by the stack base-learners stay stable after this single sample unit removal, we can argue that the marginal validity property stated in Proposition 2 still holds approximately, as will be corroborated by the empirical results in Section 4.

Suppose that in this feasible stack the first level prediction Z_{n+1} is now made by base-learners trained on the whole training sample. Following the notations introduced in Proposition 2 for the symmetric stack, let $\tilde{R}_{n+1}^{(Y_{n+1})}$ and $\tilde{R}_{(\lceil (1-\alpha)(n+1) \rceil)}^{(Y_{n+1})}$ be the corresponding random conformity scores pertaining to the feasible stack.

The following approximation result, proved in the Appendix, shows how the marginal validity of the stacked conformal prediction procedure is impacted by the broken distributional symmetry of the feasible stack: if the base-learners are stable in the sense that we can control the probability of the corresponding conformity scores on both the symmetric and the feasible stack differing too much, the marginal validity property holds approximately for the feasible stack.

Proposition 3 *Given $\epsilon > 0$, if there is a $\delta = \delta(\epsilon) > 0$ such that*

$$\Pr\left(\max\left\{\left|\tilde{R}_{n+1}^{(Y_{n+1})} - R_{n+1}^{(Y_{n+1})}\right|, \left|\tilde{R}_{(\lceil (1-\alpha)(n+1) \rceil)}^{(Y_{n+1})} - R_{(\lceil (1-\alpha)(n+1) \rceil)}^{(Y_{n+1})}\right|\right\} < \epsilon/2\right) \geq 1 - \delta,$$

then

$$\Pr(Y_{n+1} \in \tilde{C}_{n+1}^{(\alpha)}) \geq 1 - \alpha - \delta - h(\epsilon),$$

in which

$$\tilde{C}_{n+1}^{(\alpha)} = \left\{ y \in \mathbb{R} : \tilde{R}_{n+1}^{(y)} \leq \tilde{R}_{(\lceil (1-\alpha)(n+1) \rceil)}^{(y)} \right\}$$

and

$$h(\epsilon) = \Pr\left(R_{(\lceil (1-\alpha)(n+1) \rceil)}^{(Y_{n+1})} - \epsilon < R_{n+1}^{(Y_{n+1})} \leq R_{(\lceil (1-\alpha)(n+1) \rceil)}^{(Y_{n+1})}\right).$$

Algorithm 1: Full conformal prediction for multiple linear regression

Input: Matrix $Z \in \mathbb{R}^{n \times M}$ of second-level stack predictions for the training sample.
 Vector $y \in \mathbb{R}^{n \times 1}$ of training sample responses. Matrix $Z_0 \in \mathbb{R}^{m \times M}$ of
 second-level stack predictions for the size m test sample. Nominal miscoverage
 level $0 < \alpha < 1$. Tolerance $\epsilon > 0$ and multiple $u > 0$ for searching interval
 limits.
Output: Matrix $C \in \mathbb{R}^{m \times 2}$ of lower and upper prediction interval limits for the test
 sample.

```

1 sd ← standard deviation of y
2 A ← (Z⊤Z)-1
3  $\hat{\beta} \leftarrow AZ^{\top}y$ 
4 Initialize  $C = (c_{ij}) \in \mathbb{R}^{m \times 2}$  with zeroes
5 for i ← 1 to m do
6      $z_0 \leftarrow$  transpose of i-th line of  $Z_0$ 
7      $B \leftarrow A - (Az_0z_0^{\top}A)/(1 + z_0^{\top}Az_0)$ 
8      $\inf \leftarrow z_0^{\top}\hat{\beta}$ 
9     lower ← inf − u × sd
10    while inf − lower >  $\epsilon$  do
11         $y_0 \leftarrow (\text{lower} + \inf)/2$ 
12        Get scores  $r_0$  and  $\hat{r}$  inputing  $(z_0, y_0, Z, y, \alpha, A, \hat{\beta}, B)$  to Algorithm 2
13        if  $r_0 \leq \hat{r}$  then
14            | inf ←  $y_0$ 
15        else
16            | lower ←  $y_0$ 
17     $c_{i,1} \leftarrow \inf$ 
18     $\sup \leftarrow z_0^{\top}\hat{\beta}$ 
19    upper ← sup + u × sd
20    while upper − sup >  $\epsilon$  do
21         $y_0 \leftarrow (\text{sup} + \text{upper})/2$ 
22        Get scores  $r_0$  and  $\hat{r}$  inputing  $(z_0, y_0, Z, y, \alpha, A, \hat{\beta}, B)$  to Algorithm 2
23        if  $r_0 \leq \hat{r}$  then
24            | sup ←  $y_0$ 
25        else
26            | upper ←  $y_0$ 
27     $c_{i,2} \leftarrow \sup$ 
28 return C
    
```

4. Specializing the meta-learner

The feasible stack described in Section 3 is useful only if we can make the computational cost involved in the conformalization of the meta-learner manageable. It has long been known

Algorithm 2: Conformity scores computation

Input: Values of $z_0 \in \mathbb{R}^{M \times 1}$, $y_0 \in \mathbb{R}$, $Z \in \mathbb{R}^{n \times M}$, $y \in \mathbb{R}^{n \times 1}$, $0 < \alpha < 1$, $A \in \mathbb{R}^{M \times M}$, $\hat{\beta} \in \mathbb{R}^{M \times 1}$, and $B \in \mathbb{R}^{M \times M}$ from Algorithm 1.

Output: Conformity scores r_0 and \hat{r} .

```

1  $\hat{\beta}_0 \leftarrow \hat{\beta} + (y_0 - z_0^\top \hat{\beta}) B z_0$ 
2  $\hat{y} \leftarrow Z \hat{\beta}_0$ 
3  $\hat{y}_0 \leftarrow z_0^\top \hat{\beta}_0$ 
4 Initialize  $y^{(\text{res})} \in \mathbb{R}^{n \times 1}$  with zeroes
5 for  $i \leftarrow 1$  to  $n$  do
6    $y_i^{(\text{res})} \leftarrow |y_i - \hat{y}_i|$ 
7  $y_0^{(\text{res})} \leftarrow |y_0 - \hat{y}_0|$ 
8  $\hat{\beta}^{(\text{res})} \leftarrow A Z^\top y^{(\text{res})}$ 
9  $\hat{\beta}_0^{(\text{res})} \leftarrow \hat{\beta}^{(\text{res})} + (y_0^{(\text{res})} - z_0^\top \hat{\beta}^{(\text{res})}) B z_0$ 
10  $\hat{\delta} \leftarrow Z \hat{\beta}_0^{(\text{res})}$ 
11  $\hat{\delta}_0 \leftarrow z_0^\top \hat{\beta}_0^{(\text{res})}$ 
12 Initialize  $r \in \mathbb{R}^{n \times 1}$  with zeroes
13 for  $i \leftarrow 1$  to  $n$  do
14    $r_i \leftarrow y_i^{(\text{res})} / (1 + \hat{\delta}_i)$ 
15  $\hat{r} \leftarrow r_{(\lceil (1-\alpha)(n+1) \rceil)}$ 
16  $r_0 \leftarrow y_0^{(\text{res})} / (1 + \hat{\delta}_0)$ 
17 return  $(r_0, \hat{r})$ 

```

(Vovk et al., 2005) that full conformalization of a multiple linear regression model can be carried out efficiently through rank-one matrix updates made with the help of the Sherman-Morrison formula (Sherman and Morrison, 1949), which states that, if the matrix $A \in \mathbb{R}^{n \times n}$ is invertible and $u, v \in \mathbb{R}^{n \times 1}$, then $A + uv^\top$ is invertible if and only if $1 + v^\top A^{-1}u \neq 0$, in which case its inverse can be computed efficiently from A^{-1} as

$$(A + uv^\top)^{-1} = A^{-1} - \frac{A^{-1}uv^\top A^{-1}}{1 + v^\top A^{-1}u}.$$

Algorithms 1 and 2 formally describe the efficient conformalization of a multiple linear regression meta-learner based on this classical formula (see line 7 of Algorithm 1). It is worth mentioning that the residuals determination in Algorithm 2 do not involve the computation of additional inverses. The use of these residuals to construct the conformity scores denominators is crucial for obtaining prediction intervals with more adaptive width.

We implemented the stacked conformal prediction procedure for two examples. The California housing dataset (Pace and Barry, 1997) comprises information on 20,640 census tracts in California, drawn from the 1990 U.S. Census. It includes eight predictors and the response variable is the median house value in US dollars for each tract. The Ames housing dataset (De Cock, 2011) contains information on 2,930 houses sold in Ames, Iowa from 2006

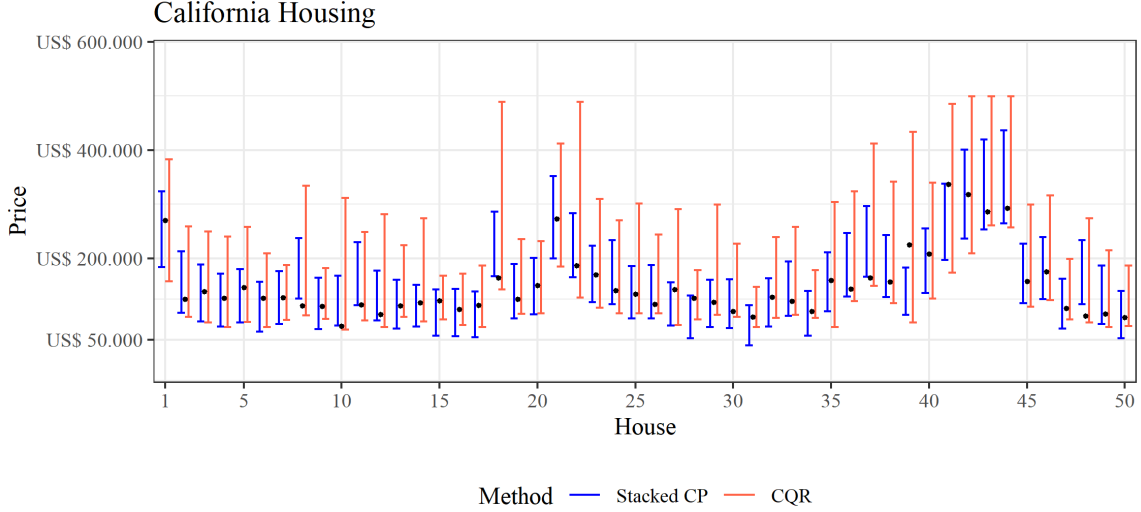


Figure 1: Prediction intervals for fifty test sample units in the California Housing dataset, using a 90% nominal coverage level ($\alpha = 0.1$).

to 2010. For the Ames dataset, we have eighty predictors and the response variable is the sale price of the house in US dollars.

We used Random Forests (Breiman, 2001) and CatBoost (Prokhorenkova et al., 2018) as the stack base-learners. For both datasets, we used 70% of the available data as our training sample, and generated prediction sets for the remaining 30% test sample units. For comparison, we used conformalized quantile regression (CQR) (Romano et al., 2019) as an inductive alternative. The Quantile Random Forest algorithm (Meinshausen, 2006) was used in the implementation of CQR. For CQR, the calibration samples sizes were chosen to be 2,000 and 500, for the California and Ames datasets, respectively, following the calibration sample size discussion in Marques F. (2025). For each dataset, Figures 1 and 2 show the prediction intervals for fifty randomly chosen test sample units, using a 90% nominal coverage level ($\alpha = 0.1$). Table 1 summarizes the results for different nominal coverage levels, showing that the stacked conformal prediction intervals exhibited appropriate empirical coverage, with shorter median interval width, when compared to the CQR results.

5. Concluding remarks and open source implementations

We have presented the stacked conformal prediction procedure in a regression setting, but its general structure readily extends to other tasks — classification being a prime example. The key requirement in any such extension is the choice of a meta-learner that admits full conformalization at manageable computational cost. In classification problems, a natural meta-learner candidate would be the k -nearest neighbors classifier, equipped with a distance metric tailored to aggregate the class-probability outputs produced by the stack

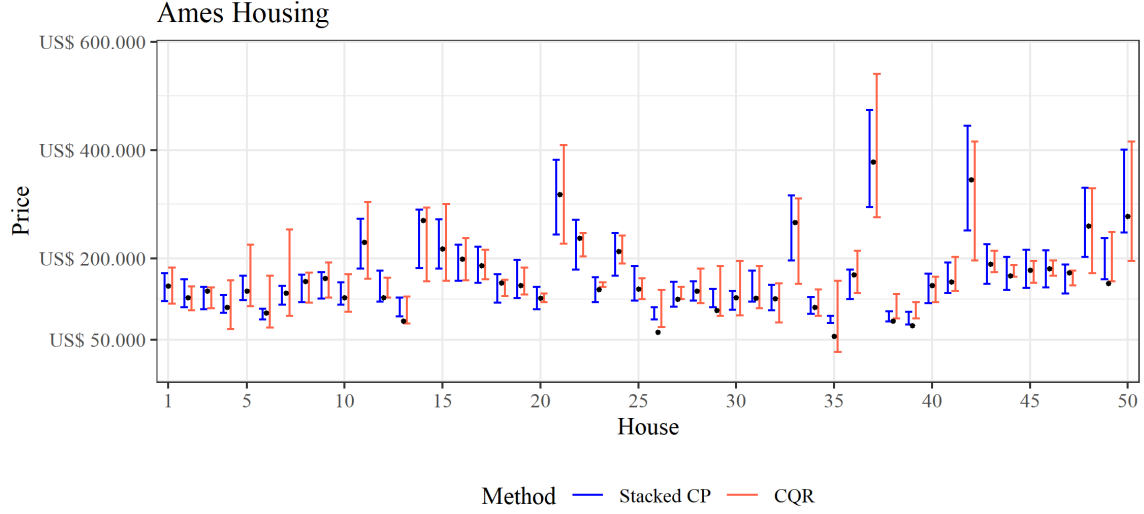


Figure 2: Prediction intervals for fifty test sample units in the Ames Housing dataset, using a 90% nominal coverage level ($\alpha = 0.1$).

Table 1: Empirical coverage and prediction interval width summaries (US dollars) for different nominal miscoverage levels α .

Dataset	Method	$1 - \alpha$	Empirical coverage	1st quartile	Median	3rd quartile
California	Stacked CP	80%	80.9%	69,226	84,995	105,696
California	CQR	80%	80.1%	67,485	105,580	162,900
California	Stacked CP	85%	85.5%	80,864	99,283	123,470
California	CQR	85%	85.7%	81,111	125,310	192,075
California	Stacked CP	90%	89.9%	96,927	119,003	147,988
California	CQR	90%	90.8%	104,433	156,600	231,500
Ames	Stacked CP	80%	81.9%	27,665	39,511	58,467
Ames	CQR	80%	81.2%	29,796	48,355	79,813
Ames	Stacked CP	85%	86.7%	32,645	46,624	69,001
Ames	CQR	85%	85.9%	34,401	55,210	91,590
Ames	Stacked CP	90%	91.1%	41,652	59,478	88,088
Ames	CQR	90%	90.2%	44,530	68,960	108,839

base-learners. R (R Core Team, 2024) and Python (Python Core Team, 2023) code, includ-

ing C++ (Stroustrup, 2013) extensions, for full reproduction of the examples in the paper are available at:

https://github.com/paulocmarquesf/stacked_cp

Acknowledgments

Paulo C. Marques F. receives support from FAPESP (Fundação de Amparo à Pesquisa do Estado de São Paulo) through project 2023/02538-0.

References

- Rina Foygel Barber, Emmanuel J. Candès, Aaditya Ramdas, and Ryan J. Tibshirani. Predictive inference with the jackknife+. *The Annals of Statistics*, 49(1):486–507, 2021. doi:[10.1214/20-AOS1965](https://doi.org/10.1214/20-AOS1965).
- Christopher M. Bishop and Hugh Bishop. *Deep learning: Foundations and concepts*. Springer Nature, 2023. doi:[10.1007/978-3-031-45468-4](https://doi.org/10.1007/978-3-031-45468-4).
- Leo Breiman. Stacked Regressions. *Machine Learning*, 24:49–64, 1996. doi:[10.1007/BF00117832](https://doi.org/10.1007/BF00117832).
- Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001. doi:[10.1023/A:1010933404324](https://doi.org/10.1023/A:1010933404324).
- Dean De Cock. Ames, Iowa: alternative to the Boston housing data as an end of semester regression project. *Journal of Statistics Education*, 19(3):1–15, 2011. doi:[10.1080/10691898.2011.11889627](https://doi.org/10.1080/10691898.2011.11889627).
- Chirag Gupta, Arun K. Kuchibhotla, and Aaditya Ramdas. Nested conformal prediction and quantile out-of-bag ensemble methods. *Pattern Recognition*, 127:108496, 2022. doi:[10.1016/j.patcog.2021.108496](https://doi.org/10.1016/j.patcog.2021.108496).
- Jing Lei, Max G’Sell, Alessandro Rinaldo, Ryan J Tibshirani, and Larry Wasserman. Distribution-free Predictive Inference for Regression. *Journal of the American Statistical Association*, 113(523):1094–1111, 2018. doi:[10.1080/01621459.2017.1307116](https://doi.org/10.1080/01621459.2017.1307116).
- Paulo C. Marques F. Universal distribution of the empirical coverage in split conformal prediction. *Statistics & Probability Letters*, 219(110350), 2025. ISSN 0167-7152. doi:[10.1016/j.spl.2024.110350](https://doi.org/10.1016/j.spl.2024.110350).
- Nicolai Meinshausen. Quantile regression forests. *Journal of Machine Learning Research*, 7(35):983–999, 2006.
- R. Kelley Pace and Ronald Barry. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297, 1997. doi:[10.1016/S0167-7152\(96\)00140-X](https://doi.org/10.1016/S0167-7152(96)00140-X).

- Harris Papadopoulos, Kostas Proedrou, Volodya Vovk, and Alex Gammerman. Inductive confidence machines for regression. In Tapio Elomaa, Heikki Mannila, and Hannu Toivonen, editors, *Machine Learning: ECML 2002*, pages 345–356, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg. ISBN 978-3-540-36755-0. doi:[10.1007/3-540-36755-1_29](https://doi.org/10.1007/3-540-36755-1_29).
- Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in Neural Information Processing Systems*, 31, 2018.
- Python Core Team. *Python: A dynamic, open source programming language*. Python Software Foundation, 2023. URL <https://www.python.org/>.
- R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2024. URL <https://www.R-project.org/>.
- Yaniv Romano, Evan Patterson, and Emmanuel Candès. Conformalized quantile regression. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 1–11. Curran Associates, Inc., 2019.
- Jack Sherman and Winifred J. Morrison. Adjustment of an inverse matrix corresponding to changes in the elements of a given column or a given row of the original matrix. *Annals of Mathematical Statistics*, 20:621–625, 1949. doi:[10.1214/aoms/1177729893](https://doi.org/10.1214/aoms/1177729893).
- Bjarne Stroustrup. *The C++ programming language*. Pearson Education, 2013.
- Vladimir Vovk. Cross-conformal predictors. *Annals of Mathematics and Artificial Intelligence*, 74:9–28, 2015. doi:[10.1007/s10472-013-9368-4](https://doi.org/10.1007/s10472-013-9368-4).
- Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer Science & Business Media, 2005. doi:[10.1007/b106715](https://doi.org/10.1007/b106715).
- Volodya Vovk, Alexander Gammerman, and Craig Saunders. Machine-learning applications of algorithmic randomness. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 444–453, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 1558606122.
- David H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992. doi:[10.1016/S0893-6080\(05\)80023-1](https://doi.org/10.1016/S0893-6080(05)80023-1).

Appendix: Proofs

Proof of Proposition 1 Laying out the second-level pairs as the $(n+1) \times (M+1)$ matrix

$$\begin{aligned}
 U = g(T, Q) &= \begin{pmatrix} Z_1 & Y_1 \\ Z_2 & Y_2 \\ \vdots & \vdots \\ Z_n & Y_n \\ Z_{n+1} & Y_{n+1} \end{pmatrix} = \begin{pmatrix} Z_{1,1} & Z_{1,2} & \cdots & Z_{1,M} & Y_1 \\ Z_{2,1} & Z_{2,2} & \cdots & Z_{2,M} & Y_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ Z_{n,1} & Z_{n,2} & \cdots & Z_{n,M} & Y_n \\ Z_{n+1,1} & Z_{n+1,2} & \cdots & Z_{n+1,M} & Y_{n+1} \end{pmatrix} \\
 &= \begin{pmatrix} \hat{\mu}_1(Q_{\setminus \phi(1)}T, X_1) & \hat{\mu}_2(Q_{\setminus \phi(1)}T, X_1) & \cdots & \hat{\mu}_M(Q_{\setminus \phi(1)}T, X_1) & Y_1 \\ \hat{\mu}_1(Q_{\setminus \phi(2)}T, X_2) & \hat{\mu}_2(Q_{\setminus \phi(2)}T, X_2) & \cdots & \hat{\mu}_M(Q_{\setminus \phi(2)}T, X_2) & Y_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \hat{\mu}_1(Q_{\setminus \phi(n)}T, X_n) & \hat{\mu}_2(Q_{\setminus \phi(n)}T, X_n) & \cdots & \hat{\mu}_M(Q_{\setminus \phi(n)}T, X_n) & Y_n \\ \hat{\mu}_1(Q_{\setminus \phi(n+1)}T, X_{n+1}) & \hat{\mu}_2(Q_{\setminus \phi(n+1)}T, X_{n+1}) & \cdots & \hat{\mu}_M(Q_{\setminus \phi(n+1)}T, X_{n+1}) & Y_{n+1} \end{pmatrix},
 \end{aligned}$$

choose any permutation matrix $\Pi \in \mathbb{S}_{n+1}$, and consider the permutation of the U rows defined by $\Pi U = \Pi g(T, Q)$. This permuted matrix $\Pi g(T, Q)$ can be obtained in a second way. Consider the folding scheme matrix defined by $Q\Pi^{-1}$. Because $(Q\Pi^{-1})(\Pi T) = QT$, this folding scheme matrix $Q\Pi^{-1}$ has the special property that it assigns the training sample units in the permuted training sample ΠT to the exact same folds that they had been assigned by the folding scheme matrix Q applied to the original training sample T . Therefore, the assumed training data symmetry (\star) of the base learners $\hat{\mu}_m$ in the definition of the symmetric stack yields the property: $\Pi g(T, Q) = g(\Pi T, Q\Pi^{-1})$. Since the class of permutation matrices is closed under multiplication and taking inverses, we have, for every $q \in \mathbb{S}_{n+1}$, that $\Pr(Q\Pi^{-1} = q) = \Pr(Q = q\Pi) = 1/(n+1)!$, implying that Q and $Q\Pi^{-1}$ have the same distribution. Because we assumed that the training sample T is exchangeable, and that T and the folding scheme matrix Q are independent, we come to the conclusion that (T, Q) and $(\Pi T, Q\Pi^{-1})$ have the same distribution. Hence, $U = g(T, Q)$ and $\Pi U = \Pi g(T, Q) = g(\Pi T, Q\Pi^{-1})$ have the same distribution, as desired. \blacksquare

The following result is used in the proof of Proposition 2.

Lemma 1 *Let $U_1, U_2, \dots, U_n, U_{n+1}$ be exchangeable random variables, and denote the order statistics of the subset $\{U_1, U_2, \dots, U_n\}$ by $V_{(1)} \leq V_{(2)} \leq \dots \leq V_{(n)}$. For each $k = 1, \dots, n$, it follows that $P(U_{n+1} \leq V_{(k)}) \geq k/(n+1)$.*

Proof Denote the order statistics of the full set $\{U_1, U_2, \dots, U_n, U_{n+1}\}$ by

$$U_{(1)} \leq U_{(2)} \leq \dots \leq U_{(n)} \leq U_{(n+1)}.$$

Considering that we may have ties, it follows from the definition of $U_{(k)}$ that at least k of the U_i 's are less than or equal to $U_{(k)}$, for $k = 1, \dots, n+1$. Hence, $\sum_{i=1}^{n+1} I_{\{U_i \leq U_{(k)}\}} \geq k$, almost surely. Taking expectations and observing that, by exchangeability, the probability $P(U_i \leq U_{(k)})$ is the same, for $i = 1, \dots, n+1$, we have that $P(U_{n+1} \leq U_{(k)}) \geq k/(n+1)$. Furthermore, for $k = 1, \dots, n$, notice that $U_{n+1} > V_{(k)}$ if and only if $U_{n+1} > U_{(k)}$, because U_{n+1} cannot be strictly larger than itself. Hence, $P(U_{n+1} \leq V_{(k)}) = P(U_{n+1} \leq U_{(k)})$, for $k = 1, \dots, n$, and the result follows. \blacksquare

Proof of Proposition 2 Taking $y = Y_{n+1}$, the definitions in the proposition statement and the result in Proposition 1 imply that the random conformity scores

$$R_1^{(Y_{n+1})}, R_2^{(Y_{n+1})}, \dots, R_n^{(Y_{n+1})}, R_{n+1}^{(Y_{n+1})}$$

are exchangeable. Using Lemma 1 above, we have that

$$\Pr\left(R_{n+1}^{(Y_{n+1})} \leq R_{(k)}^{(Y_{n+1})}\right) \geq \frac{k}{n+1},$$

for $k = 1, \dots, n$. Choosing $k = \lceil (1 - \alpha)(n + 1) \rceil \geq (1 - \alpha)(n + 1)$, the result follows, since the definition of the random prediction set $C_{n+1}^{(\alpha)}$ entails that $Y_{n+1} \in C_{n+1}^{(\alpha)}$ if and only if $R_{n+1}^{(Y_{n+1})} \leq R_{(\lceil (1 - \alpha)(n + 1) \rceil)}^{(Y_{n+1})}$. \blacksquare

Proof of Proposition 3 Introduce the notations

$$r = R_{(\lceil (1 - \alpha)(n + 1) \rceil)}^{(Y_{n+1})} \quad \text{and} \quad \tilde{r} = \tilde{R}_{(\lceil (1 - \alpha)(n + 1) \rceil)}^{(Y_{n+1})},$$

and define the event

$$B_\epsilon = \left\{ \max \left\{ \left| \tilde{R}_{n+1}^{(Y_{n+1})} - R_{n+1}^{(Y_{n+1})} \right|, |\tilde{r} - r| \right\} < \epsilon/2 \right\}.$$

If the statement in B_ϵ is true and $R_{n+1}^{(Y_{n+1})} \leq r - \epsilon$, then $\tilde{R}_{n+1}^{(Y_{n+1})} < R_{n+1}^{(Y_{n+1})} + \epsilon/2 \leq r - \epsilon/2 < \tilde{r}$. Therefore, using the lower bound in Proposition 2, the result follows:

$$\begin{aligned} \Pr\left(Y_{n+1} \in \tilde{C}_{n+1}^{(\alpha)}\right) &= \Pr\left(\tilde{R}_{n+1}^{(Y_{n+1})} \leq \tilde{r}\right) \\ &\geq \Pr\left(\left\{\tilde{R}_{n+1}^{(Y_{n+1})} < \tilde{r}\right\} \cap B_\epsilon\right) \\ &\geq \Pr\left(\left\{R_{n+1}^{(Y_{n+1})} \leq r - \epsilon\right\} \cap B_\epsilon\right) \\ &= \Pr\left(R_{n+1}^{(Y_{n+1})} \leq r - \epsilon\right) - \Pr\left(\left\{R_{n+1}^{(Y_{n+1})} \leq r - \epsilon\right\} \cap B_\epsilon^c\right) \\ &\geq \Pr\left(R_{n+1}^{(Y_{n+1})} \leq r - \epsilon\right) - \Pr(B_\epsilon^c) \\ &\geq \Pr\left(R_{n+1}^{(Y_{n+1})} \leq r - \epsilon\right) - \delta \\ &= \Pr\left(R_{n+1}^{(Y_{n+1})} \leq r\right) - \Pr\left(r - \epsilon < R_{n+1}^{(Y_{n+1})} \leq r\right) - \delta \\ &= \Pr\left(Y_{n+1} \in C_{n+1}^{(\alpha)}\right) - \Pr\left(r - \epsilon < R_{n+1}^{(Y_{n+1})} \leq r\right) - \delta \\ &\geq 1 - \alpha - \delta - \Pr\left(r - \epsilon < R_{n+1}^{(Y_{n+1})} \leq r\right) \\ &= 1 - \alpha - \delta - h(\epsilon). \end{aligned}$$

\blacksquare