Applying Conformal Prediction to the Loan Approval Process of a German Bank

Jonas Fassbender jonas@fassbender.dev

Abstract

This paper investigates the applicability of conformal prediction for enhancing the loan approval process of a German bank. In recent years comparison portals for consumer loans have established themselves as an easy-to-use way for consumers to find the best conditions for their loan requests. The comparison portals have led to unprecedented amounts of loan requests. This results in dropping rates of successful completion, leading to reduced profits for banks. Banks have to query information about the consumer making a request from credit bureaus in order to assess the risk of default and consequently approve or decline the request. Querying information from a credit bureau costs money. Banks are therefore striving to filter out bad requests likely to be declined, before they query information from a credit bureau. This paper describes a novel approach for implementing such a filter with a restricted abstaining classifier based on an adaptation of the standard conformal prediction algorithm. This model fits the problem at hand better, where we have two disadvantages compared to other problems conformal prediction is applied to: (i) we cannot leverage the validity of a conformal predictor directly, because a conformal predictor does not guarantee validity for a single prediction set. (ii) we deal with an online framework where we need a fast prediction for an incoming loan request. Conformal prediction is computationally expensive. Our adaptation only approximates p-values to generate fast predictions and fast updates of the predictor with new examples. Because we cannot leverage validity directly we look at two weaker properties for assessing the reliability of our model. We conduct an experiment on the data of a German bank to see whether these weaker properties hold and to see if our approach offers a viable solution for implementing a filter for bad loan requests.

Keywords: Reliable Machine Learning, Conformal Prediction, Abstaining Classifiers, Finance

1. Introduction

In recent years comparison portals have established themselves as a convenient way for consumers to find the bank giving them the best conditions for their loan request. Banks who sell such consumer loans are partnering up with the comparison portals and sell their products over this new platform, reaching more potential customers. Unfortunately, this new way of reaching potential customers puts the banks under financial stress concerning the decline of the ratio of successfully completed loan requests. For a consumer, making a loan request is free of charge. On the other hand, the decision process of the bank for

accepting or declining such a request is not. The bank has to collect information about the consumer from a credit bureau in order to assess the risk of default. Getting the information about the consumer cost the bank money, so with a decline of the ratio of successfully completed loans, costs are getting higher.

A consumer—when making a loan request—fills out a web form with personal information, like his income or how many children she has. In order to reduce the cost for the loan approval process, banks are trying to implement filters which identify bad requests based on the data from the web form, before information is queried from the credit bureau. This reduces the overall cost of the loan approval process.

Reliability for such a filter is key, since approved loan requests wrongfully classified by the filter as being likely declined will cost the bank as well, because the bank loses the chance of successfully completing the loan request. The conformal prediction framework with its natural approach to reliability seems like a great fit for implementing a reliable filter.

Unfortunately, the problem of implementing such a filter is rather constraining, which is why we had to differ from the standard conformal prediction approach in some ways. The most significant constraint for our approach is the fact, that the validity of a conformal predictor (see Section 2) can not be directly leveraged by the filter, because conformal prediction is not conditionally valid over every prediction set. Validity is only averaged over all prediction sets combined. If one is only interested in a subset of all prediction sets, the predictor does not have to be valid. For this reason, two weaker properties for achieving certainty in the predictions of the filter are proposed: (i) the accuracy of the filter should approximate the confidence level (see Section 2) and (ii) the accuracy of the filter should be consistent over time. The second property being the more important one for assessing the reliability of the filter in the online setting.

The other constraint is the semi-online context of the problem, which offers us only a lazy teaching schedule and time constraints for both predicting and updating operations of our model (for a definition of teaching schedules see Vovk et al., 2005). The time constraint led us to using an adaptation of the standard conformal predictor, which is computationally more efficient but less precise than conformal prediction and inductive conformal prediction in the semi-online context. The adaptation achieves its better computational performance by only approximating p-values.

Our experiment on the data provided by the German bank shows, that depending on the significance level ϵ and the laziness of the teacher, we can achieve a lesser, but reasonable amount of confidence in our model, without it having to achieve the much stricter validity, by at least fulfilling the weaker property of consistency over time. Our approach looks much more like classical supervised machine learning with its hyperparameter tuning approach for finding well calibrated models.

This paper continues by giving a short introduction on what conformal prediction is and outlines the aspects of it which are important for this paper in Section 2. Afterwards we have a look at our proposed approach in Section 3. We begin by giving a high level

overview over the loan approval process of the bank whose data we used for our experiment presented in Section 5. Then we go into detail on how the problem constraints us and the way we deal with these constraints. The two weaker properties we want our model to fulfill in the absence of validity are presented, as is the computationally more efficient adaptation of a conformal predictor which only approximates the p-values used for prediction. Section 4 describes the dataset used for the experiment. The experiment (Section 5) analyzes different models on how well they achieve the weaker properties. These results and in general the applicability of the proposed approach are discussed in Section 6, before at last a conclusion is drawn in Section 7.

2. Conformal Prediction

This section gives an outline of how conformal prediction (CP) works and how it is applied in the context of supervised learning. It focuses on the aspects that are important for this paper. The section also describes inductive conformal prediction (ICP)—a computationally less expensive adaptation of CP—and how one can achieve conditional validity by splitting the example space into categories using conditional (I)CP.

Let $\mathbf{Z}$ be our example space, where each example $z \in \mathbf{Z}$ is a tuple (x, y). $x \in \mathbb{R}^d$ is called the observation, while $y \in \mathbf{Y}$ is called the label of example z. Since we are interested in classification rather than regression, $\mathbf{Y}$, the label space, is a finite set. CP differs from common classification, which aims to predict the label of a new and unseen observation x_{n+1} based on a model derived from previously seen examples $\{z_1, \ldots, z_n\}$, the training set. The model most commonly returns a score for each possible label from the label space and the label generating the highest score is returned as the prediction y_{n+1}.

A conformal predictor can be described as a confidence predictor Γ^ϵ, where ϵ is called the significance level. Γ^ϵ returns a prediction set $\Gamma^\epsilon_{n+1}(x_{n+1}) \subseteq \mathbf{Y}$, which, as long as exchangeability holds for $\mathbf{Z}$, contains the true label of x_{n+1} with a probability of one minus ϵ. One minus ϵ is called the confidence level (Vovk et al., 2005). The conformal predictor is based on a nonconformity measure A, which returns a nonconformity score for an example. The nonconformity score is an indication of how well the example fits into the set of previously seen examples. In order to generate the prediction set for x_{n+1}, the conformal predictor extends the set of previous seen examples with a provisional example z'_{n+1}, which consists of x_{n+1} mapped to an element y' of $\mathbf{Y}$:

$$z'_{n+1} := (x_{n+1}, y'). \tag{1}$$

The examples from the extended set are then mapped to nonconformity scores α_i with a nonconformity measure A:

$$\alpha_i := A(\{z_1, \ldots, z_n, z'_{n+1}\}, z_i), i = 1, \ldots, n \tag{2}$$

$$\alpha_{n+1} := A(\{z_1, \ldots, z_n, z'_{n+1}\}, z'_{n+1}). \tag{3}$$

In order for the conformal predictor to achieve validity, it is not enough to just look at the nonconformity score α_{n+1} of z'_{n+1}, especially since different A can return nonconformity scores on different scales. Instead, α_{n+1} is looked at in the context of all other nonconformity scores $\alpha_1, \ldots, \alpha_n$. For this, the p-value of z'_{n+1} is computed:

$$p_{z'_{n+1}} := \frac{|\{i = 1, \ldots, n : \alpha_i \geq \alpha_{n+1}\}|}{n+1}. \tag{4}$$

$p_{z'_{n+1}}$ is computed for every label y' from the label space $\mathbf{Y}$. The prediction set Γ^{ϵ}_{n+1} contains every label which produces a p-value that is bigger than ϵ:

$$\Gamma^{\epsilon}_{n+1}(x_{n+1}) := \{\forall y' \in \mathbf{Y} : p_{z'_{n+1}} > \epsilon\}. \tag{5}$$

Nonconformity measures are normally based on so called underlying algorithms which are common machine learning algorithms for classification, like k-nearest neighbors, neural networks, support vector machines or random forests (see e.g. Vovk et al., 2005; Balasubramanian et al., 2014; Papadopoulos et al., 2007; Devetyarov and Nouretdinov, 2010).

CP can be applied in an online and semi-online framework, as it is in this paper. It is often referred to as transductive conformal prediction, where no general rule is inductively derived from the training set before x_{n+1} is predicted. Depending on the underlying algorithm—which very well can be an inductive algorithm—and for large training sets, CP can be computationally very expensive (Vovk et al., 2005).

2.1 Inductive Conformal Prediction

If one looks at a conformal predictor described above, it is clear that computing the nonconformity scores for each example from the training set for each label from the label space every time a new observation should be predicted is not computationally feasible for large training sets and for underlying algorithms that are computationally expensive to train, like neural networks (Papadopoulos et al., 2007). For large data sets and in the context of underlying algorithms that inductively derive a prediction rule from the training set, ICP was designed to address the problem of the high demands of computation.

For ICP the training set is split into the proper training set $\{z_1, \ldots, z_m\}$ and the calibration set $\{z_{m+1}, \ldots, z_n\}$, $m < n$. The underlying algorithm is trained only once on the proper training set. The nonconformity scores $\alpha_{m+1}, \ldots, \alpha_{n+1}$ are computed like:

$$\alpha_i := A(\{z_{m+1}, \ldots, z_n\}, z_i), i = m+1, \ldots, n \tag{6}$$

$$\alpha_{n+1} := A(\{z_{m+1}, \ldots, z_n\}, z'_{n+1}). \tag{7}$$

The p-value for z'_{n+1} is again computed like (4), with i ranging from $m+1$ to n and the divisor being $n-m+1$ (Balasubramanian et al., 2014). Validity under the exchangeability assumption holds for ICP like it does for CP, as long as it is used in at least a semi-online framework. ICP's validity is weakened, if it is used as a purely offline predictor.

The predictive efficiency of an inductive conformal predictor can suffer in exchange for less computation (Vovk et al., 2005). Vovk et al. (2019) provides different criteria for the predictive efficiency of an (inductive) conformal predictor. The criteria used in this paper can be found in Section 3.2.

2.2 Conditional Conformal Prediction

Conditional CP is most interesting for asymmetric problems where, for example, the cost of misclassifying label y^1 as label y^2 is much higher than misclassifying label y^2 as label y^1. Validity in itself is only an average over all predictions the conformal predictor makes. Vovk et al. (2005) gives an example for a conformal predictor used on the USPS handwritten digits dataset with a confidence level of 95%. While the predictor was valid over every prediction it made, it produced an error rate of 11.7% for every example with a true label of "5", far below its confidence level.

(I)CP can overcome this averaging over every prediction regardless of its category (in the case above the category would be the label of the example) by constructing a conditional (inductive) conformal predictor. As long as exchangeability holds, the conditional predictor will fulfill the stronger conditional validity. This means that the error probability for any category will not exceed ϵ (see e.g. Vovk et al., 2005; Balasubramanian et al., 2014; Vovk, 2013).

Let $\mathbf{K}$ be the finite set of all categories and $K : \mathbf{Z} \to \mathbf{K}$ a function that maps an example to its category. The p-value for the example z'_{n+1} is now computed as

$$p_{z'_{n+1}} := \frac{|\{i = 1, \ldots, n : K(z_i) = K(z'_{n+1}) \;\&\; \alpha_i \geq \alpha_{n+1}\}|}{|\{i = 1, \ldots, n : K(z_i) = K(z'_{n+1})\}| + 1}, \tag{8}$$

for a conformal predictor. The p-value for a conditional inductive conformal predictor is computed like (8), only i changes like described in Section 2.1. Label-conditional (I)CP is the special case, where $\mathbf{K} = \mathbf{Y}$ and $K(z_i) = y_i$.

3. Proposed Approach

This chapter describes the loan approval process on an abstract level and gives some background on why a filter mechanism which removes bad loan requests early in the approval process is desired. Afterwards the model used to implement the filter is described. Time constraints and the semi-online framework of the problem domain are the key to the design decisions made for implementing the model. The deviations from the standard (I)CP framework to deal with the environment are outlined and discussed. The performance criteria used in assessing the model in the following chapters is given and the fact that for this problem validity must be disregarded in favor of weaker properties for assessing reliability is discussed.

3.1 The Loan Approval Process

In recent years German comparison portals like `www.check24.de` have established themselves as a common way for consumers who are seeking a loan to find the bank which offers them the best conditions. Banks compete on these portals for customers and unprecedented amounts of loan requests are made. For consumers making a loan comparison is free of charge. The consumer fills out a web form containing information about his income and other personal information. This web form is sent to every bank which has partnered up with the comparison portal. The banks then go through their loan approval processes, where the loan request is either approved or declined. If the loan request is approved, the bank will be listed with the conditions for the requested loan in the browser of the consumer who then can decide which conditions—if any—benefit him the most. Afterwards the consumer can take any of the listed offers.

This new form of comparing consumer loans put banks under a lot of pressure concerning the declining ratio of successfully completed loan requests. While requesting a comparison of conditions on the comparison portal is free of charge for the consumer, the loan approval process of the bank is not. The bank has to query information about the person making the request at a credit bureau, which costs the bank money. If the successfully completed loans coming through the comparison portal do not increase proportional to all the loan requests coming from it, the bank will ultimately lose money. This state is not beneficial for any participant. Some banks will—in the long run—abandon the comparison portals. This may result in consumers missing out on the most beneficial conditions they could get, leading to consumers abandoning the comparison portals as well. This means the comparison portals will lose their commissions for successfully completed loan requests over their platform.

Banks therefore are striving to implement filters. A filter is a decision rule which works on the form a consumer submits. It tries to filter out bad loan requests based on the data, so it can decline them before making the costly request to the credit bureau, just to decline the request afterwards. Reliability of such a filter is key, since potentially approved loan requests which are wrongfully filtered out as bad requests will lead to profit demise. This makes the CP framework a perfect fit for such a complex problem domain.

Figure 1 displays a high level overview over the loan approval process of the bank whose data is used for the experiment presented in Section 5. The first step after the request is received is the validation of the fields of the submitted form. If any field does not pass this validation stage (e.g. negative income), the form is considered wrong and the request is declined early.

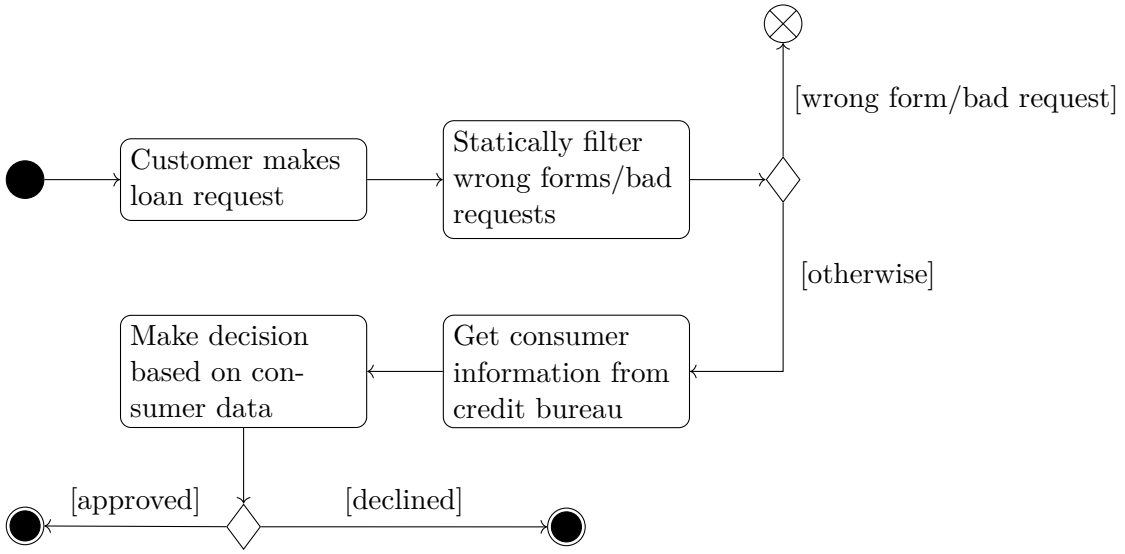


Figure 1: UML activity diagram giving a high level overview over the current loan approval process.

The bank already reacted to its declining ratio of successful loan requests by implementing a static filter. After a loan request has passed the validation stage it goes through this static filter and is either declined right away—the request is considered bad, likely to be declined—or the process continues if the filter considers the request possible for approval. In that case, information about the consumer are collected from the credit bureau and based on this a final decision is made. For liability reasons only bad requests can be declined beforehand. Good ones cannot be accepted before going through the stage where information comes from the credit bureau.

The model outlined in the next chapter should enhance the loan approval process by replacing the static filter with a dynamic filter evolving over time and learning from the incoming requests. Of course, instead of replacing the static filter with the dynamic one, both can be used at the same time by applying them in succession to an incoming loan request.

3.2 The Model

This chapter will focus on the changes made to the CP framework in order to fit it best to the above described loan approval process which is rather restrictive by providing two main constraints: the performance criteria and time.

prediction \ true label	positive	negative
positive	True Positive (**TP**)	False Positive (**FP**)
negative	False Negative (**FN**)	True Negative (**TN**)
?	Rejected Positive (**RP**)	Rejected Negative (**RN**)

Table 1: Confusion matrix for a binary abstaining classifier.

Let $\mathbf{Y} = \{accepted, declined\}$ be the binary label space of the problem this paper discusses. All possible subsets $2^{\mathbf{Y}}$ of $\mathbf{Y}$ are:

$$2^{\mathbf{Y}} = \{\{\}, \{accepted\}, \{declined\}, \{accepted, declined\}\}. \tag{9}$$

A prediction set for an unseen observation x_{n+1} is any set from $2^{\mathbf{Y}}$. For the dynamic filter, only the prediction set $\{declined\}$ is of any interest, because then we can say with certainty one minus ϵ, that the true label of x_{n+1} is indeed *declined* and we can exit the process early without having to spend money on getting consumer information from the credit bureau. For any other prediction set, the model should forward x_{n+1} to the next step of the process. On top of our conformal predictor Γ^ϵ we can build the filter:

$$\phi^\epsilon(x_{n+1}) := \begin{cases} declined & \text{if } \Gamma^\epsilon(\{z_1, \ldots, z_n\}, x_{n+1}) = \{declined\} \\ ? & \text{otherwise.} \end{cases} \tag{10}$$

ϕ^ϵ is a restricted abstaining classifier. An abstaining classifier is a classifier that can either return a prediction from $\mathbf{Y}$ or abstain from making a prediction if it is not certain enough (indicated by the ? element) (see e.g. Friedel, 2005; Friedel et al., 2006; Smirnov et al., 2009). ϕ^ϵ is restricted in the sense that a normal abstaining classifier can return an element from the whole set $\mathbf{Y} \cup \{?\}$, while in this case ϕ^ϵ can only return a subset of $\mathbf{Y}$ or the ? element.

Table 1 shows the confusion matrix of a binary abstaining classifier. It is the same as for non-abstaining binary classifiers, just with the rejected positives RP and rejected negatives RN added to the matrix. Based on the confusion matrix we can derive our two main criteria for the performance of ϕ^ϵ: its accuracy Acc and its efficiency E. We can define both measurements Acc and E as:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} = \frac{TN}{TN + FN} \tag{11}$$

$$E = \frac{TN}{RN + TN + FP} = \frac{TN}{RN + TN}. \tag{12}$$

Acc is normally defined as all true predictions TP plus TN divided by all predictions made. If we set negative to be equal to the *declined* element of $\mathbf{Y}$, we can remove TP and

FP from consideration, since ϕ^ϵ is not able to predict the positive label (*accepted*). TP and FP will be equal to zero. The same goes for E, which is the amount of TN predictions divided by all elements having negative as their true label. E gives an indication on how many of all the declined loan requests are actually detected by the filter.

Even when exchangeability holds, ϕ^ϵ must not be valid: $Acc(\phi^\epsilon)$ being greater or equal to the significance level one minus ϵ. Unfortunately, validity is not conditional over the different prediction sets. The problem—like for the general unconditional validity discussed in Section 2.2—is that validity is just an average over every prediction set returned by the conformal predictor. If one is only interested in a subset of all possible prediction sets, the accuracy of the conformal predictor can be lower than the confidence level. In this case, if $\Gamma^\epsilon_{n+1} = \{accepted, declined\}$, the true label of x_{n+1} is guaranteed to be in the prediction set, since it is equal to $\mathbf{Y}$. Every time the conformal predictor predicts the whole label space it cannot make a mistake. This can conceal its below validity performance on other prediction sets.

An example which illustrates this dilemma can be found in Shafer and Vovk (2008). Table 3 in Shafer and Vovk (2008) shows the results of an experiment with three different conformal predictors on the iris dataset, where the classifiers should distinguish between the two species iris setosa (s) and iris versicolor (v) based on the sepal length. While all three classifiers are valid, only one is valid over all singleton prediction sets ($\{s\}$ and $\{v\}$). The other two produce an average below the predefined confidence level of 0.92.

Why still use a computationally demanding conformal predictor as the underlying method for ϕ^ϵ, instead of any normal scoring classifier? The idea is that while validity cannot be exploited directly, it still gives an indirect way of effectively tuning the model. While ϕ^ϵ may not be valid, it hopefully still fulfills two weaker properties: (i) the accuracy of ϕ^ϵ approximates the significance level (one minus ϵ) and (ii) the accuracy of ϕ^ϵ is consistent over time. The latter property is the stronger one, because it gives certainty in future predictions of the model even in the absence of validity. If both properties hold for the problem at hand will be shown and discussed in Section 5.

Another aspect which makes conformal prediction a well suited underlying algorithm for ϕ is the fact that ϵ is well defined in the sense that its implications for the performance of the model are easily understood. If ϵ is very small, the model will be more conservative compared to a risk-taking model with a bigger ϵ. Furthermore CP fits well to the problem at hand, which provides a semi-online context.

Using CP without leveraging its validity directly makes the whole process of finding a well calibrated classifier for the problem at hand much more relatable to the trial and error or hyperparameter tuning approach used by more common supervised learning methods, which do not make any guarantees to their validity. In this context ϵ can be thought of as a tunable hyperparameter. Finding the right set of hyperparameters for calibrating an algorithm well is not a trivial task. Most algorithms provide a lot of hyperparameters (like deep neural networks) or their hyperparameters come from a very complex space (the kernel parameter for kernelized algorithms like support vector machines or Gaussian processes).

Having just a single, powerful hyperparameter coming from a simple space like ϵ makes searching for a well calibrated model much easier and less time consuming than calibrating most scoring classifiers. Furthermore, CP is basically a way for making normal scoring classifiers reliable, since any scoring classifier can be used as the underlying algorithm for the nonconformity measurement.

The second constraint mentioned above was time. The process shown in Figure 1 is timed out. If the bank can not process a request fast enough, it will not be displayed with its conditions in the consumer's web browser, even if the request gets accepted. The model therefore is constrained by the time it is allowed to spend on computing its prediction. While ICP deals with exactly this time constraint on prediction time, it does not provide a fast update mechanism needed in this semi-online context where we want our model to be updated with the outcome of every request it abstained from predicting. This is called a lazy teaching schedule (Vovk et al., 2005).

While it is easy to increase the calibration set of the inductive conformal predictor (just add the nonconformity score of the new example to the nonconformity scores of the other elements of the calibration set), it is not trivial to extend the proper training set. After extending the proper training set, the underlying algorithm has to be trained which can be a costly operation depending on the algorithm. Furthermore the nonconformity scores of the calibration set must be updated, an operation that can take time for a big calibration set. While such an operation could be done infrequently in an asynchronous fashion—a copy of the classifier is updated and replaces the classifier used for prediction after the updating process is finished—the result would be that incoming requests are not processed by an up-to-date predictor. This is a state we are aiming to avoid.

In the following chapters, a computationally less expensive adaptation of CP—in the semi-online context of the problem at hand—was used as the underlying model for ϕ^ϵ. Every time the underlying algorithm is updated with an example z_{n+1}, it returns the nonconformity score for it based on the set of its predecessors $z_1, \ldots, z_n$. The nonconformity score α_{n+1} is added to the sequence of nonconformity scores $\alpha_1, \ldots, \alpha_n$ used to generate the p-value for the next incoming observation x_{n+2}. The sequence $\alpha_1, \ldots, \alpha_{n+1}$ is computed like:

$$(\alpha_1, \alpha_2, \ldots, \alpha_{n+1}) = (A(\{\}, z_1), A(\{z_1\}, z_2), \ldots, A(\{z_1, z_2, \ldots, z_n\}, z_{n+1})). \tag{13}$$

The proposed model disregards ever computing a nonconformity score with either (2) or (3) in favor of only using (7) with m equal to zero for every element of the training set. The nonconformity scores $\alpha_1, \ldots, \alpha_n$ are basically frozen in time and never updated. For every update of our model, we only have to compute one nonconformity score instead of having to compute n plus one scores. This gives a computational advantage for bigger n. The idea is that after a certain n the nonconformity scores—and consequently the p-values generated from them—normalize and α_n will not differ much—if at all—from its true value (if it would be updated with (2) once a new example is added to the training set). The accuracy of the p-value is disregarded in favor of a less computationally expensive model.

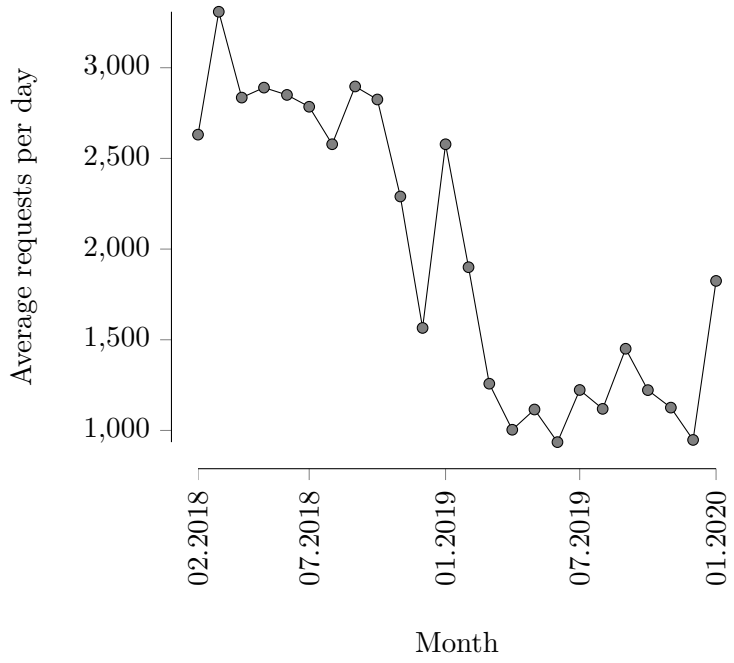


Figure 2: The average requests per day for every month the dataset covers. In March of 2019, the static filter was deployed, which accounts for the decrease of requests. The amount of requests per day which went through the whole decision process drops from an average of approximately 2600 requests per day to approximately 1200 requests per day after the deployment.

4. The Dataset

The dataset used for the experiment presented in the following chapter consists of fields of the web form the consumer submits when making a loan request and fields derived from that form, e.g. age from the difference between the date of the request and the birth date submitted in the form. The data is anonymized. The consumers making the loan requests can not be identified based on the dataset.

The label is the result of the decision process shown in Figure 1. The dataset has 111 features and contains approximately 1.35 million examples which were collected between the 25th of February 2018 and the 20th of January 2020. A timespan of 693 days. The dataset is sparse. Over half of the values are missing.

The main characteristic of the dataset is the deployment of the static model in March of 2019, which filters bad requests likely to be declined by the decision process. Figure 2 shows the average incoming requests per day in every month while Figure 3 shows the

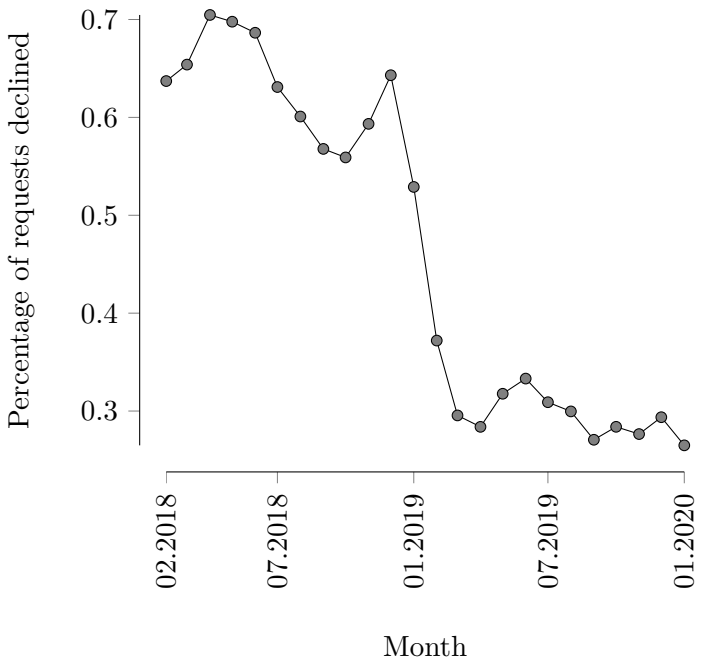


Figure 3: The percentage of requests declined in each month the dataset covers. After the model was deployed, the overall percentage of approximately 60% declined requests drops to approximately 30% of declined requests which went through the whole decision process.

percentage of declined requests in every month in the dataset. Requests filtered by the static filter are not part of the dataset. Both figures clearly show a drop in incoming requests and the percentage of declined requests beginning in March 2019.

Out of the 693 days the dataset contains, 368 days are before the 1st of March 2019, amounting to approximately 53% of the time span covered by the dataset. This time period contains approximately 70% of all the examples while of these examples approximately 60% were rejected. Since March 2019 only 30% of the examples are rejected. Overall approximately 52% of all examples contained in the dataset are rejected.

5. Experiment

This chapter describes the conducted experiment which should show the applicability of the above described model ϕ^ϵ to the dataset described in Section 4. It will discuss if the two weaker properties we want ϕ^ϵ to achieve in the absence of validity hold. The two properties are: (i) the accuracy of the model should approximate the confidence level (one minus ϵ)

and (ii) the accuracy of the model should be consistent over time. The latter being the stronger property concerning the reliability of the model. The dataset has a unique and for a model challenging characteristic: in March of 2019 a static filter was deployed which changes the distribution of the data (see Section 4). Since we basically have to deal with two different datasets, this chapter will also concern itself with analyzing the results based on the presence and absence of the static filter—below denoted as Φ_s. The used methods will be briefly outlined before the main results are presented.

Like stated in Section 4: the dataset is sparse. In order to deal with the sparsity of the dataset, we reduced its dimensionality using PCA.[1] The data was reduced to ten dimensions. The nonconformity measure used is based on the 1-nearest neighbor method extensively covered by various literature about conformal prediction (see e.g. Vovk et al., 2005; Balasubramanian et al., 2014; Shafer and Vovk, 2008). A nonconformity score α_{n+1} is computed as:

$$A(\{z_1, \ldots, z_n\}, (x_{n+1}, y')) = \frac{\min_{i=1,\ldots,n:y_i=y'} d(x_i, x_{n+1})}{\min_{i=1,\ldots,n:y_i\neq y'} d(x_i, x_{n+1})}, \tag{14}$$

$d(x_i, x_{n+1})$ being the L_1 distance between x_i and x_{n+1}. The conformal predictor ϕ^ϵ is based on is label conditional (see Section 2.2).

Additional to ϵ, we added a second hyperparameter for tuning ϕ^ϵ: feedback. While Section 3.2 states that we want to update ϕ^ϵ only with every abstained prediction, we also tested with giving the model feedback over a few of its predictions, revealing the true label to the model. A certain percentage of predicted observations ($\phi_i^\epsilon = \textit{declined}$) still go through the decision process. The teaching schedule is less lazy. Tested were a teaching schedule with zero feedback (only abstained predictions are used for updating), ten percent and 100% feedback. 100% feedback—while in practice useless—represents the results in the pure online setting. The experiment tried four different ϵ: 0.001, 0.005, 0.01 and 0.025.

Table 2 shows the overall results of the experiment while Figure 4 shows both performance criteria *Acc* and E over the timespan the model was applied to. Neither adjusts the feedback when computing E which would always be zero for the 100% feedback. For example the model with $\epsilon = 0.005$ and ten percent feedback has an overall efficiency of 0.22, which means that 22% of all the declined requests are actually filtered by the model. If the feedback is taken into account, the efficiency of this model drops to 20% overall.

The experiment clearly reveals that property (i) does not hold. Only for the most conservative model ($\epsilon = 0.001$ and 100% feedback) does the accuracy approximate one minus ϵ. The accuracy of all other, more risk taking models does not approximate one minus ϵ and the more risk the model takes the less approximate the accuracy is. The fact that (i) does not hold for all ϵ is most clearly shown by the most risk-taking model tested ($\epsilon = 0.025$ and zero percent feedback) which can be seen as an outlier (see Table 2). While for all other models more risk taking ones are less accurate and more efficient, the

1. For this we used the implementation from the scikit-learn library (Buitinck et al., 2013).

ϵ		Feedback: 0.0			Feedback: 0.1			Feedback: 1.0		
		$\neg\Phi_s$	Φ_s	Σ	$\neg\Phi_s$	Φ_s	Σ	$\neg\Phi_s$	Φ_s	Σ
0.001	*Acc*	0.97	0.95	0.97	0.98	0.98	0.98	0.99	1.0	0.99
	E	0.18	0.14	0.17	0.17	0.16	0.17	0.14	0.17	0.15
0.005	*Acc*	0.89	0.77	0.87	0.94	0.93	0.94	0.97	0.97	0.97
	E	0.22	0.17	0.21	0.22	0.21	0.22	0.21	0.24	0.21
0.01	*Acc*	0.83	0.66	0.8	0.9	0.85	0.89	0.96	0.95	0.96
	E	0.25	0.19	0.24	0.26	0.23	0.25	0.23	0.29	0.24
0.025	*Acc*	0.51	0.3	0.47	0.83	0.75	0.82	0.92	0.91	0.92
	E	0.27	0.18	0.26	0.3	0.31	0.3	0.29	0.41	0.31

Table 2: The results of the conducted experiment. For each tested ϵ and for each amount of feedback, the two performance criteria *Acc* and E are displayed. The results are analyzed based on whether the static filter was active (Φ_s) or not ($\neg\Phi_s$). The Σ columns give the performance criteria over the whole dataset, regardless of whether the filter was active or not.

most risk-taking one has an accuracy far less than all other models but does not increase its efficiency. This indicates that ϵ can not be arbitrarily defined and only generates reasonable results in certain intervals and with a certain amount of feedback/laziness of the teacher.

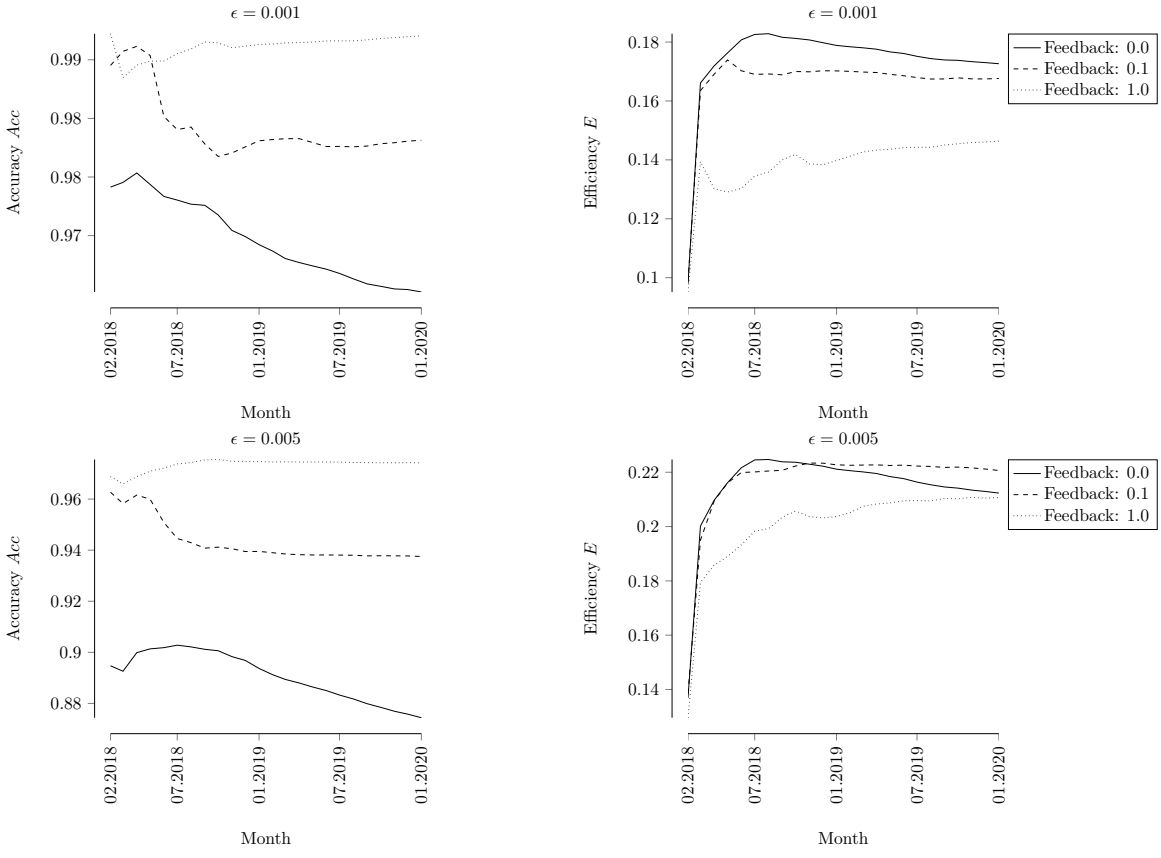
The accuracy of the models does not approximate the significance level. The first property is not fulfilled in any context but the fully online one with the most conservative model tested. On the other hand property (ii) does hold for all models with feedback (see Figure 4). Again the accuracy of more risk taking models is less consistent over time like it is less approximate to one minus ϵ.

Any model with 100% feedback is consistent over time. The accuracy of the models with ten percent feedback converges at a certain accuracy once it has seen more examples (see Figure 4). It does not drop, after the deployment of the static model. The accuracy of the models with zero percent feedback on the other hand drops over time. Revealing all abstained examples to a model with zero percent feedback while concealing the examples it predicted seem to bias it to increase its certainty in uncertain regions. It makes the model more confident but less reliable.

The efficiency of the models with ten percent feedback reaches a constant point, unlike the efficiency of the models with zero percent feedback which drops together with their accuracy. The efficiency of models with 100% feedback—if one does not consider that their efficiency is actually zero—keeps increasing over the time span covered by the dataset.

Surprisingly, the deployment of the static filter does not seem to influence any model. Models with feedback tend to have a consistent accuracy regardless of the static filter, having converged before the deployment (see Figure 4; deployment was in March 2019).

Even more surprising is the consistency of the efficiency of all models. Models with 100% feedback continue to increase their efficiency after the deployment. The efficiency of models with ten percent feedback neither increases—because the amount of declined and abstained requests drops—nor decreases, because patterns based on which the model found bad requests are removed by the static filter. Models with zero percent feedback start their decline in accuracy and efficiency before the deployment of the static filter.



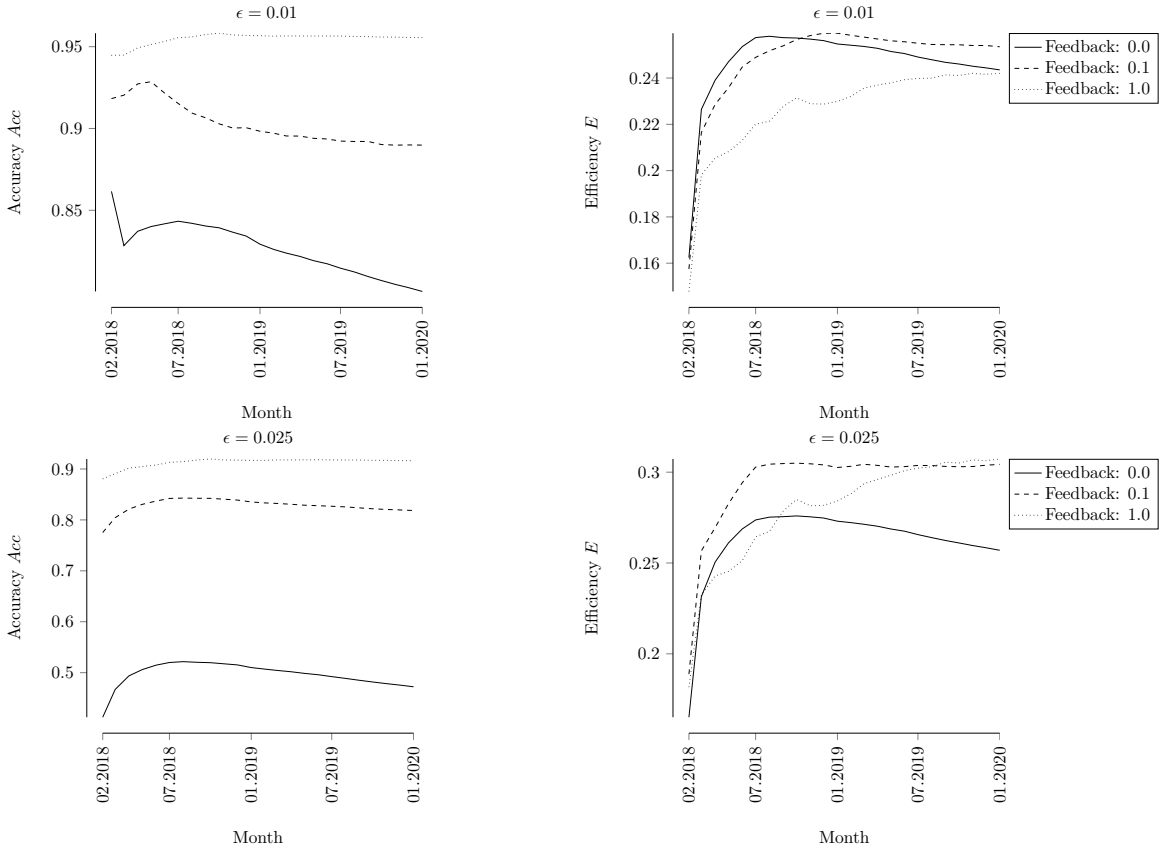


Figure 4: Time series plots on how both performance criteria *Acc* and E behave for each constellation of ϵ and feedback tested. All models with a feedback of zero percent have an early peak, before declining both in accuracy and efficiency. They do not fulfill consistency over time. With ten percent feedback *Acc* and E are consistent after a certain amount of examples are revealed. 100% feedback also generates consistent *Acc*, but its efficiency is still increasing.

6. Discussion

This paper deals with a constrained problem domain (see Section 3). The constraints are:

1. The context of the problem is semi-online with a lazy teacher.

2. Prediction and updating the model must both be fast operations.

3. We only want to predict on a certain prediction set which means we can not rely on the validity of our model.

4. The dataset is sparse and parts of it are filtered by a static filter which changes the distribution of the dataset.

To cope with those constraints we defined an abstaining classifier ϕ^ϵ on top of a computationally less expensive adaptation of CP. Since we can not leverage the validity of CP directly we defined two weaker properties which were tested in the previous chapter. While neither of the two weaker properties hold for all of the twelve tested models, the stronger one—consistency over time—was achieved by all models with feedback, giving a reasonable amount of certainty in absence of validity for these models.

In general, the proposed approach looks much more like conventional supervised machine learning with its hyperparameter tuning approach for finding well calibrated models. In this context we considered two powerful hyperparameters—ϵ and the amount of feedback the lazy teacher discloses—for calibrating ϕ^ϵ. A surprising finding of the experiment was that the static filter does not seem to influence either accuracy or efficiency of the tested models. The experiment also shows that ϵ alone is not enough for tuning ϕ^ϵ. The teaching schedule must be considered as well. A lazy teacher who just discloses abstained predictions leads to models that decline both in accuracy and efficiency after a certain amount of time, violating the property of consistency over time.

It must be said that the experiment presented in this paper concerns itself with just the first prototype of a solution for the problem of dynamically predicting, whether a loan request will be declined or not. While the main approach for the model is at center and will likely be augmented by further work, no other work was done at this time concerning other parameters. The dataset is not yet analyzed and no work was done for cleaning and feature selection/generation besides dimensionality reduction using PCA. Furthermore a very profound part of ϕ^ϵ, the underlying algorithm, is not considered at all. Only the 1-nearest neighbor method presented in the previous chapter was ever tried as nonconformity measurement. Especially in our setting with the weakened properties instead of validity, the underlying algorithm for the nonconformity measurement is much more important.

While this means that the presented problem most certainly has a better solution than one of the models tested, the novel approach for ϕ^ϵ and its underlying, computationally more efficient adaptation of CP which only approximates p-values, still provides a powerful framework for future work, both in augmenting ϕ^ϵ further and for finding a better fitting model for predicting declined loan requests. An example for how ϕ^ϵ could be augmented would be to minimize a loss function over ϵ and to dynamically set the amount of feedback based on the current performance of the model. While the presented adaptation of CP never updates nonconformity values that were generated in the past, it could periodically update a constant number of them which means it would still have a constant time complexity of $\mathcal{O}(1)$ (if the time complexity of generating a single nonconformity score is disregarded) and be more precise. Testing the difference of the CP adaptation used here against the original CP setting will also be an emphasis in continuing the work on the problem at hand.

7. Conclusion

For a constraining problem—reliably predicting the outcome of the loan approval process of a German bank—we have defined a restricted abstaining classifier ϕ^ϵ on top of a computationally more efficient, less precise adaptation of a conformal predictor. Since a conformal predictor achieves validity unconditionally to the prediction sets, we can not exploit validity directly. In the absence of validity we have defined two weaker properties for assessing the reliability of ϕ^ϵ and did an experiment to see whether these two properties hold. Without validity our approach resembles much more that of classical supervised machine learning algorithms concerning the calibration of a model: hyperparameter tuning. In this context we looked at two powerful hyperparameters for calibrating ϕ^ϵ: ϵ and the amount of feedback the lazy teacher discloses. With both parameters one can achieve our second property: consistency over time, making ϕ^ϵ reasonably reliable without it having to achieve validity directly. Our approach simply takes more work calibrating ϕ^ϵ compared to a normal conformal predictor which is valid under exchangeability, regardless of its parameters.

While we deem the conducted experiment successfully shows that ϕ^ϵ is a good approach for enhancing the loan approval process, it just represents the first prototype of our efforts. The dataset is still a mystery, as is the underlying algorithm for the nonconformity measure and how it effects the reliability of ϕ^ϵ. Also ϕ^ϵ can be improved further. We will continue our work and try to improve it further. Two ideas for improvement are outlined in Section 6.

Using abstaining classification the way it is in this paper will hopefully be a building block for others who try to enhance a process with a reliable classifier. It provides a fast and easy way for integrating machine learning into a process and makes gradual deployment possible when we do not trust our model to fully replace the process. Or where it is not possible to do so like in this case for liability reasons. If ϕ^ϵ or the computationally more efficient adaptation of CP is useful for other types of constraining problems where the classical CP approach may not be applicable remains to be seen. Lastly, hopefully this paper will provide the CP community—or in general people who are interested in reliable machine learning—with valuable insights into the problems of a complex process from a field where reliability is key and where CP most certainly will establish itself in the near future as one of the main ways for achieving reliability: finance.

References

Vineeth Balasubramanian, Shen-Shyang Ho, and Vladimir Vovk. *Conformal Prediction for Reliable Machine Learning: Theory, Adaptations and Applications*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, first edition, 2014. ISBN 0123985374, 9780123985378.

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert

Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013.

Dmitry Devetyarov and Ilia Nouretdinov. Prediction with confidence based on a random forest classifier. volume 339, pages 37–44, 10 2010. doi: 10.1007/978-3-642-16239-8_8.

Caroline Friedel. On abstaining classifiers. 01 2005.

Caroline C. Friedel, Ulrich Rückert, and Stefan Kramer. Cost curves for abstaining classifiers. In *Proceedings of the ICML 2006 workshop on ROC Analysis in Machine Learning*, 2006.

Harris Papadopoulos, Vladimir Vovk, and Alex Gammerman. Conformal prediction with neural networks. volume 2, pages 388 – 395, 11 2007. ISBN 978-0-7695-3015-4. doi: 10.1109/ICTAI.2007.47.

Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9:371–421, jun 2008. ISSN 1532-4435.

Evgueni Smirnov, Georgi Nalbantovi, and A. M. Kaptein. Meta-conformity approach to reliable classification. *Intelligent Data Analysis*, 13, 01 2009. doi: 10.3233/IDA-2009-0400.

Vladimir Vovk. Conditional validity of inductive conformal predictors. *CoRR*, 2013. URL `http://alrw.net/articles/05.pdf`.

Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, 2005.

Vladimir Vovk, Valentina Fedorova, Ilia Nouretdinov, Ivan Petej, and Alexander Gammerman. Criteria of efficiency for conformal prediction. *CoRR*, 2019. URL `http://alrw.net/articles/11.pdf`.