Rethinking and Recomputing the Value of Machine Learning Models

Burcu Sayin[1], Jie Yang[2], Xinyue Chen[2], Andrea Passerini[1], and Fabio Casati[3]

[1] University of Trento, Via Sommarive, 9, Povo TN, Italy
[2] Delft University of Technology, Mekelweg 5, Delft, Netherlands
[3] Servicenow, Santa Clara 9, CA, USA

Abstract. In this paper, we argue that the prevailing approach to training and evaluating machine learning models often fails to consider their real-world application within organizational or societal contexts, where they are intended to create value for people. By adopting this perspective, we fundamentally change how we evaluate and select machine learning models. Specifically, we focus on the implementation of machine learning models into real-world workflows involving both machines and human experts, with the latter being involved whenever the machine is not confident enough in its prediction and abstains. We show how standard quality metrics like accuracy and f-score are inappropriate in measuring the real value of machine learning models in such hybrid human-machine settings. To mitigate this problem, we introduce a simple but theoretically sound strategy to adapt existing machine learning models so as to maximize value. An extensive experimental evaluation highlights the importance of the value-based perspective in evaluating models and the impact of calibration and out-of-distribution settings on model value.

Keywords: hybrid intelligence · selective classification

1 Introduction

Recently, a few position papers [11, 54, 55] have challenged the underlying assumptions of quality in Machine Learning (ML), particularly the overemphasis on accuracy-based metrics and various measures of calibration errors. There are two observations at this stance: (i) ML models are almost always applied in hybrid human-machine settings, where the model can abstain or reject predicting if the confidence is insufficient (Fig. 1), and (ii) the value (cost) of correct/incorrect inferences or rejections is determined by the use case, not by the model.

We have experienced that the majority of AI deployments in the enterprise consist of selective models or *selective classifiers* [23], which is more of a rule than an exception. An example where this commonly occurs is in customer support requests, where the goal is to identify the customer's intent to trigger an automated request processing workflow if possible. Failing to comprehend

Corresponding author: burcu.sayin@unitn.it

the customer's intent and resorting to human agents is not ideal. However, it is even more problematic to misinterpret the customer's intent and guide them down the wrong path toward a resolution. This is why intent classifications are filtered based on prediction confidence. How "good" or "useful" a model is therefore depends on the value it brings to the ML solution workflows (Fig. 1). This value depends on how often the workflow rejects the predictions, on the correctness patterns of the predictions that are not rejected, and on the "cost" of errors vs. benefits of correct predictions. This *value* notion is not what model accuracy or F1 score measure. While in this paper we only marginally discuss the plethora of Large Language Models (LLMs), the problem is exactly the same if not worse: With generative AI the question of whether to show an answer or to withhold it is crucial, and there are many things that can be wrong in an answer (e.g. hallucination). We add that the fact that some APIs do not reveal likelihood/confidence level makes model evaluation more difficult.

To some extent, all this is trivial. There is no inherent difficulty in developing use case-based value functions, selecting the best model from a set of well-performing models based on the value function, or evaluating a model's performance across multiple value functions. Moreover, one could contend that accuracy metrics are a sufficient substitute for evaluating model improvements in data science, or for selecting models to deploy in an AI platform designed to meet specific use cases. Thus, a practical approach would be to select the model with the highest accuracy or F1 score and allow users to filter out low-confidence predictions. Accuracy metrics are easy to understand and don't require determining parameters like "cost of errors," which can be challenging to estimate.

In this paper, we show that this reasoning is wrong. If we accept that classifiers are mostly applied as selective models, then the method we use to measure, compare, and even train models must change. The implications of models being almost always applied as selective classifiers are often neglected in the literature, and this is also reflected in model leaderboards. We also show that the simplicity of not having to choose a cost parameter is an illusion: when we use accuracy to

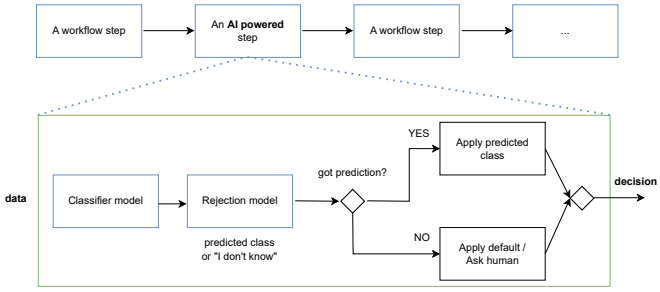

Fig. 1: Typical implementation of ML models into an ML solution workflow. The rejection function filters based on a confidence threshold, assuming that the classifier is trained independently of the rejection logic. This does not have to be - the classifier can be aware of the cost, in which case it becomes less "general".

compare models, i) we do implicitly choose a cost parameter, often without realizing it, and ii) this implicitly selected cost is probably one of the worst choices: that of setting the relative cost of errors to zero. Despite being counter-intuitive, we show that accuracy is a quality metric that may be selected when the consequences of model errors are not critical. When a model is likely to be used across multiple use cases, relying solely on accuracy-based metrics can have significant implications. Overall, we show that: (i) Universal metrics used for model evaluation are poor indicators of model value, potentially leading to incorrect decisions such as choosing models with *negative* value, (ii) Metrics designed to account for cost-sensitive errors are also inappropriate as they fail to consider the reject option, (iii) Lack of calibration substantially affects model value, and poorly calibrated complex models can be outperformed by simple, decades-old models that are easier to calibrate, and (iv) Operating in an out-of-distribution (OOD) setting further reduces the reliability of standard performance metrics.

The remainder of this paper is structured as follows: in Section 2, we review related works on our concept of model value. Then, in Section 3, we formalize this notion and introduce the rejection threshold maximizing value, along with its extension to the cost-sensitive setting where different errors have different costs. Section 4 presents our experimental analysis comparing our value metric with standard performance measures, while Section 5 offers our conclusions.

2 Related work

Selective classification. Mimicking the typical use of ML models in many practical applications, a number of approaches rely on the combination of an ML model making an initial prediction and a human annotator taking over when the model's confidence is not high enough [10]. Selective classifiers are specifically conceived for this use, by including a rejection mechanism to decide when to abstain from making a prediction. The literature on selective classifiers is extensive, encompassing a broad range of learning algorithms, including nearest-neighbor classifiers [30], SVM [21], and neural networks [14, 15, 23] (see Hendrickx et al. [31] for a recent survey). The effectiveness of this solution is, however, heavily dependent on the reliability of machine confidence, which has shown to be very poor, especially for deep learning [5, 24].

Classifier confidence. To effectively use a classifier [32], it is important to understand its properties and have confidence in its individual predictions. The literature proposes various confidence-based methods, including measuring the entropy of the softmax predictions [61], calculating trust scores based on the distance of samples to a calibration set [32], determining a confidence threshold (via Shannon entropy [56], Gini coefficient [6], or norm-based methods [45]) that maximizes coverage for a given accuracy [9], and using semantics-preserving data transformation to estimate confidence [4]. Post-hoc recalibration is a popular strategy for improving classifier confidence, with techniques ranging from temperature scaling [24] to Dirichlet calibration [36] (see a recent survey by Filho et al. [58]). However, as we will show in our experimental evaluation (see Sec-

tion 4.2), it's essential to complement these solutions with a proper value metric to assess the classifier's utility in real-world applications.

Cost-sensitive learning tackles classifier training challenges by accounting for different error costs, especially in scenarios with imbalanced classes [19, 39, 62, 65]. Existing work has [65]: (i) data-level approaches [63, 72] where the class distribution of training data is balanced via sampling methods, and (ii) algorithm-level approaches, that use a thresholding scheme [12, 18, 19, 39, 53, 57, 59] to improve the prediction performance on the minority class (e.g. in binary classification, the threshold is set such that the prediction is 1 only if the expected cost associated with this prediction is lower than or equal to that of predicting 0). Although this field is closely related to our setting as it considers the impact of errors on the downstream pipeline, it assumes that the classifier provides a prediction for every instance without any rejection mechanism. This assumption can significantly impact the evaluation of the resulting classifier's quality, as our experimental evaluation will demonstrate (see Section 4.2).

Hybrid Human-AI systems aim at solving classification problems with humans and machines [16, 17, 50, 68], but effectively combining human and machine intelligence has many challenges. For example, *trust in humans* requires a deep understanding of how to design crowdsourcing tasks and model their complexity [22, 47, 69, 71], test and filter crowd workers [7], aggregate results into a decision [25, 34, 35, 38, 40, 67, 73], improve the engagement [26, 27, 48], or leverage crowds to learn features of ML models [13, 51]. Furthermore, *the effective aggregation of human and machine decisions* [43, 44, 46] depends on many factors, such as training, explaining, sustaining, interacting, and amplifying. We believe that defining appropriate measures of the value of the joint human-machine system is a major prerequisite to keeping research in the field on the right course.

3 Measuring model "value"

In this section, we formally define the notion of model "value", and show how threshold-based selective classifiers, by far the most popular class of classifiers in practical ML workflows, can be adjusted to maximize value.

3.1 The setting

Selective classifiers can be implemented as follows: (a) We take a model f that outputs a prediction y and a *confidence* c_y (or a vector $\mathbf{c}$ of confidence for a set of possible answers). Then, we filter the predictions to take only those above a certain confidence threshold (Fig. 2a), (b) f outputs predictions and confidence, but a selector model s determines whether to accept the prediction by considering input features x (Fig. 2b), (c) A hybrid of the two above cases is where the selector is a recalibrator r that can either take as input the prediction and confidence measure (*feature-agnostic* calibrator) or also the input features of x and adjust the confidence vector (*feature-aware* calibrator), typically applying threshold-based selection on the resulting confidence (Fig. 2c), and (d) The

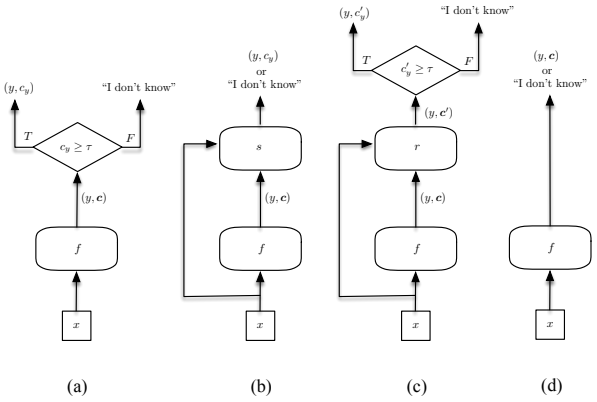

Fig. 2: Typical ways of selectivity in classification.

model f is already trained to only output predictions that are "good enough" and includes an "I don't know" class (Fig. 2d).

The first case is the most common, at least in our experience. The second case is an extension and generalization of the first case in two ways: it can take features as input (s can be trained as opposed to "just" being a formula), and it can filter based on any formula. It however requires some form of "training" or machine teaching, which is highly non-trivial. The recalibrator also typically requires some form of training. However, a feature-agnostic calibrator can be easily set up by posthoc calibration strategies [58] (e.g. temperature scaling [24]). Finally, the last case is what is being addressed by the recent literature on learning to reject [31], which is currently confined to the academic world, but it could highly benefit by incorporating the notion of value that we introduce here.

In formalizing "value", we will progressively make a few assumptions that i) allow to simplify the presentation of the problem without altering the essence of the concepts, ii) are reasonable in many if not most use cases, and iii) make the definition of the value function easier to understand and interpret for the users who eventually have to deploy ML into their companies. We scope the conversation on classification problems as it makes it easy to ground the examples and terminology, and because it is easier to define a notion of accuracy. This is important: people understand accuracy because it is simple, and that has value even if accuracy is "inaccurate" as a metric, and most users will not be able to express complex value functions. Note however that our results also apply to other metrics (e.g. F1-score) as we will show in our experimental evaluation.

3.2 Definition of value

We have a classifier g that operates on items $x \in \mathcal{D}$ and returns either a predicted class $y \in \mathcal{Y}$ or a special label y_r, denoting "rejection" of the prediction. Then,

the average value per the prediction of applying a model g over $\mathcal{D}$ becomes:

$$V(g, \mathcal{D}) = \rho V_r + (1 - \rho)(\alpha V_c + \Sigma_{ij}[\Omega \odot V_w]_{ij}) \tag{1}$$

where ρ is the proportion of items in $\mathcal{D}$ that are rejected by g (classified as y_r), α is the accuracy for predictions above the threshold, V_r and V_c are the value of rejecting an item and classifying it correctly respectively, Ω is a matrix denoting the proportion of predictions (above threshold) in each cell of the confusion matrix, and V_w is a matrix with the cost for each type of error (set to zero on the main diagonal corresponding to correct predictions), and $\odot$ denotes the Hadamard (element-wise) product, of which we take the summation across all elements ij. Notice that ρ, α, Ω all depend on $\mathcal{D}$ and g, and we omit the indices to simplify notation. Also, if our classification problem has $|\mathcal{Y}|$ classes, then Ω and V_w are $|\mathcal{Y}| \times |\mathcal{Y}|$ (y_r is not included here). An alternative representation would be to just say that $V(g, \mathcal{D}) = \Omega' V'$, where the confusion and value matrices incorporate the reject class. This would allow us to model class-dependent values of rejections and correct predictions. Instead, if we only consider costs based on what we misclassify then Ω and V_w become vectors, and in the most common case where all wrong predictions are considered equally bad in a first approximation, then Ω and V_w are scalar, and $\Omega = 1 - \alpha$, so in this case, the formula becomes:

$$V(g, \mathcal{D}) = \rho V_r + (1 - \rho)(\alpha V_c + (1 - \alpha)V_w) \tag{2}$$

At this point, we simplify the notation to remove dimensionality (values can be measured in dollars, but here we care about relative values because we want to compare models and learning strategies), and arrive at a formulation that is digestible for process owners (the people who apply AI in their processes), for whom it may be hard to come up with the three cost parameters/vectors. None of the above simplifications change the concepts presented.

We define as baseline the case where we do not have ML, or, equivalently, where we reject any prediction. We set this baseline at 0 ($V_r = 0$); making it easier to evaluate a model in terms of (i) whether it improves on the baseline or not, and (ii) whether we should adopt AI for a given problem.

$$V(g, \mathcal{D}) = (1 - \rho)(\alpha V_c + (1 - \alpha)V_w) \tag{3}$$

We also express V_w in terms of V_c, as in $V_w = -kV_c$, where k is a constant telling us how bad is an error with respect to getting the correct prediction:

$$V(g, \mathcal{D}) = V_c(1 - \rho)(\alpha - k(1 - \alpha)) \tag{4}$$

V_c is a scaling factor for the above value formula. When reasoning about an AI-powered solution workflow we disregard that factor, we can think in terms of value "per unit of V_c dollars", or equivalently assume the magnitude of V_c, so we can focus on value. From now on we, therefore, focus on "value per dollar unit of rejection cost" $V' = V/V_c$. We avoid introducing a new symbol and, without loss of generality with respect to the above equations, we set $V_c = 1$ to get Equation 5 capturing the same concepts as Equation 1 but simplifying our presentation.

$$V(g, \mathcal{D}) = (1 - \rho)(\alpha - k(1 - \alpha)) \tag{5}$$

3.3 Filtering by threshold

We now focus on the most common situation observed in practice; the model selectivity is applied by thresholding confidence values and rejecting predictions that have confidence c less than a threshold τ (case (a) in Figure 2). We are given a model m that processes items $x \in \mathcal{D}$ and returns a vector of confidences (one per class). This is the output of a softmax; for each x, we consider the pair $\hat{y}, \hat{c}$ corresponding to the top-level prediction of $m(x)$ and the confidence associated with the prediction. Given a threshold τ, we define a function s:

$$s(\hat{y}, \hat{c}, \tau) = \begin{cases} \hat{y} & \text{if } \hat{c} \geq \tau, \\ y_r & \text{otherwise.} \end{cases}$$

where y_r is the special class label denoting "rejection" of the prediction. Our classifier g is therefore now expressed in terms of m and τ. This means that we can express the value as a function of $m, \mathcal{D}, \tau$. In a given use case, when we are given m and have knowledge of Ω (or of k in the simplified case), we select the threshold $\tau \in [0, 1]$ that optimizes $V(g, \mathcal{D})$ (We assume τ is unique or we randomly pick one if not). Thus, we can express the value of our classification logic as a function of $(m, \mathcal{D}, k)$:

$$V(m, \mathcal{D}, k) = (1 - \rho_\tau)(\alpha_\tau - k(1 - \alpha_\tau)) \tag{6}$$

Notice that τ can be set empirically on some tuning dataset $\mathcal{D}$ (it depends on $m, \mathcal{D}, k$), and ρ_τ and α_τ reflect the proportions ρ and α given τ. However, if we are aware of the properties of confidence vectors, we can set τ regardless of $\mathcal{D}$. For example, if we assume perfect calibration (where the expected accuracy for a prediction of confidence c is c) [58], then we know that the threshold is at the point where the value of accepting a prediction is greater than zero and $\alpha_\tau = \tau$. This means that to have $V(m, \mathcal{D}, k) > 0$ we need $\tau - k + k\tau > 0$, which means

$$\tau > k/(k + 1) \tag{7}$$

This conforms to intuition: if k is large, it never makes sense to predict, better go with the default. If k=0 (no cost for errors), we might always predict since there is no penalty for applying inaccurate predictions. Perhaps paradoxically, this case where inaccurate predictions are harmless is when *accuracy* is the metric we want to use. If k=1 (errors are the mirror image of correct predictions), then our threshold is 0.5. Figure 3(a) shows how a simple threshold-based selector can be adapted to maximize model value.

We assumed that all errors have equal cost when deriving the threshold, but it can be extended to cost-sensitive settings, as demonstrated next.

3.4 Cost-sensitive value and thresholds

In this section, we extend the discussion on the value and optimal threshold to the setting in which different errors have different costs (and possibly, different

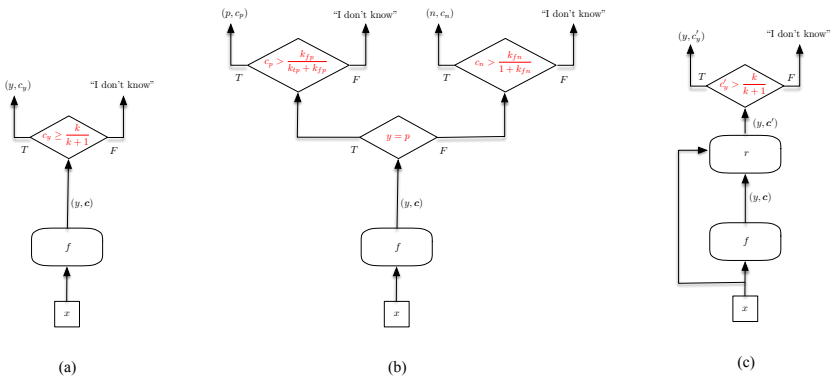

Fig. 3: Adapting selective classifiers to maximize value: (a) threshold-based selector (b) cost-sensitive threshold-based selector; (c) recalibrator + threshold-based selector. Changes with respect to standard counterparts are highlighted in red.

correct predictions have different values). We focus on the binary classification setting for simplicity, but the reasoning can be easily generalized to multiclass classification. The cumulative value of a selective classifier g on a dataset $\mathcal{D}$ can be written as (setting $V_r = 0$ as in the cost-insensitive case):

$$V(g, \mathcal{D}) = (1 - \rho)(N_{tp}V_{tp} + N_{tn}V_{tn} + N_{fp}V_{fp} + N_{fn}V_{fn})$$

where $N_{tp}, N_{tn}, N_{fp}, N_{fn}$ are the numbers of true positives, true negatives, false positives, and false negatives in $\mathcal{D}$, and $V_{tp}, V_{tn}, V_{fp}, V_{fn}$ are the values associated to the corresponding predictions. Let V_c be the base cost for a correct prediction. This is typically associated with a correctly predicted negative instance, i.e., $V_{tn} = V_c$. We can define the other values as multiples of this base cost as follows:

$$V_{tp} = k_{tp}V_c, \quad V_{fp} = -k_{fp}V_c, \quad V_{fn} = -k_{fn}V_c$$

for some user-defined and application-specific constants k_{tp}, k_{fp}, k_{fn}. The cumulative value simplifies as:

$$\begin{aligned} V(g, \mathcal{D}) &= (1 - \rho)(N_{tp}k_{tp}V_c + N_{tn}V_c - N_{fp}k_{fp}V_c - N_{fn}k_{fn}V_c) \\ &= (1 - \rho)V_c(k_{tp}N_{tp} + N_{tn} - k_{fp}N_{fp} - k_{fn}N_{fn}) \end{aligned}$$

Setting $V_c = 1$ (unit of value) as in the cost-insensitive case, we get:

$$V(g, \mathcal{D}) = (1 - \rho)(k_{tp}N_{tp} + N_{tn} - k_{fp}N_{fp} - k_{fn}N_{fn})$$

Let's now focus on the standard setting of a classifier rejecting by threshold. Note that we need to set class-specific thresholds τ_p and τ_n for positive and negative predictions respectively to account for the different costs. Consider an instance x predicted as positive by the classifier. It's expected value (according

to the predictions in $\mathcal{D}$) is given by:

$$\begin{aligned} V(g,x) &= (1-\rho)(k_{tp}N_{tp}/N_p - k_{fp}(N_{fp}/N_p) \\ &= (1-\rho)(k_{tp}N_{tp}/N_p - k_{fp}(1-N_{tp}/N_p)) \\ &= (1-\rho)(N_{tp}/N_p(k_{tp}+k_{fp}) - k_{fp}) \end{aligned}$$

where we normalized N_{tp} and N_{fp} by N_p, the number of positive instances in $\mathcal{D}$, to turn them into probabilities, and we removed the terms containing N_{tn} and N_{fn} as their corresponding probabilities are zero if the instance is predicted as positive. If the classifier is perfectly calibrated, we know that $N_{tp}/N_p = \tau_p$. A positive value for the instance is thus achieved by setting τ_p as:

$$\tau_p > \frac{k_{fp}}{k_{tp}+k_{fp}} \tag{8}$$

If x is predicted as negative by the classifier, it is expected value is given by:

$$\begin{aligned} V(g,x) &= (1-\rho)(N_{tn}/N_n - k_{fn}N_{fn}/N_n) \\ &= (1-\rho)(N_{tn}/N_n - k_{fn}(1-N_{tn}/N_n)) \\ &= (1-\rho)(N_{tn}/N_n(1+k_{fn}) - k_{fn}) \end{aligned}$$

where N_n is the number of negative instances in the training set. If the classifier is perfectly calibrated, we know that $N_{tn}/N_n = \tau_n$. A positive value for the instance is thus achieved by setting τ_n as:

$$\tau_n > \frac{k_{fn}}{1+k_{fn}} \tag{9}$$

Figure 3(b) shows how to adjust a threshold-based selector to maximize value in a cost-sensitive setting. We assumed a binary classification setting for the sake of simplicity, but the derivation can be easily extended to account for class-specific thresholds in multiclass classification.

4 Experiments

We now show how a value-oriented perspective affects the model evaluation and use. In particular, we try to answer the following questions: **Q1)** Is model accuracy (or F1-score) a sensible indicator of the value of a model?, **Q2)** Is cost-sensitive error a sensible indicator of the value of a model in cost-sensitive settings?, **Q3)** How does calibration affect the value of a model?, and **Q4)** How does predicting in an OOD setting affect the value of a model?

Our experimental evaluation is focused on NLP classification tasks, for which we analyze the behavior of simple as well as state-of-the-art models over various datasets, models, and text encoders. This choice stems from the broad diffusion of NLP models in companies, and from our experience in industrial use cases that were all NLP-based. However, the concept of value can be applied to any

ML model deployed in a practical application, and we believe that the main results of our experimental evaluation hold for many other domains. We refer the reader to our GitHub repo[4] for the companion code.

4.1 Experimental Setup

Datasets and Tasks We use the following datasets in our experiments. *Hate-speech detection on Twitter:* We replicated the original tests from [2] by analyzing two widely used models ([1, 3]) and testing them on the Waseem et al. [66] dataset. However, we could only recover 9671 of the tweets as of October 2021 (the original dataset size is 14949). *Clickbait detection:* The *Clickbait Challenge* on the *Webis Clickbait Corpus 2017*[5] was classifying Twitter posts as a clickbait or not. Both training and test sets are publicly available[6], while each team was free to choose a subset of the training set for validation (we followed the "blobfish" team). *Multi-Domain Sentiment Analysis - and Dataset (MDS):* Sentiment analysis based on a dataset for domain adaptation[7]. The data includes four categories of Amazon products (DVD, Books, Electronics, and Kitchen). The task is to learn sentiment from one of these domains and test it on the others.
Models and text encoders. We use various models for each task. For the hate-speech dataset, we test the following leaderboard models: (i) *Badjatiya et al.* [3] which uses an RNN to construct word embeddings and then classify them with Gradient-Boosted Decision Tree. In the original paper, test accuracy is measured as the average of the ten folds in cross-validation; however, in our reproduction, we separated validation and test set before cross-validation, and they are used for evaluation only after training. (ii) one model from *Agrawal and Awekar* [1] which is composed of an embedding layer followed by a Bidirectional LSTM and a fully connected layer with softmax activation. For the clickbait detection dataset, we test 4 models from one leaderboard team on clickbait challenge: *fullnetconc*, *weNet*, *lingNet*, and *fullNet* which are published on Github[8]. This team modified the task into binary classification - they categorized items with a score under 0.5 into "non-clickbaiting", and vice versa. For the MDS dataset, we referred to the leaderboard for the sentiment analysis task of Domain adaptation[9] and tested the best-performing leader-board model, *Multi-task tri-training (mttri)* by Ruder et. al. [52], that leverages multi-task learning strategies to improve the performance of tri-training. As the source code of other competing approaches was not publicly available, we compared mttri with three baseline models from the scikit-learn library[10]: (i) a simple Logistic Regression model (*LogR*); (ii) a basic MLP with a single hidden layer (*MLP1*); (iii) an MLP with four hidden layers ("*MLP4*). All models were tested with a simple *TF-IDF* encoding.

[4] https://tinyurl.com/rethinking-value-of-ml-models
[5] https://webis.de/data/webis-clickbait-17.html
[6] https://zenodo.org/record/5530410#.YWcFtC8RrRV
[7] http://nlpprogress.com/english/domain_adaptation.html
[8] github.com/clickbait-challenge/blobfish
[9] nlpprogress.com/english/domain_adaptation.html
[10] https://scikit-learn.org/

Table 1: Comparison of accuracy, F1, and value for increasing values of k.

Task	Model	Acc	F1	Value for k=?				
				1	2	4	8	10
Hate Speech	Badj. et al	**0.82**	**0.63**	**0.64**	**0.5**	**0.36**	**0.27**	**0.22**
	Agr. et al.	0.73	0.62	0.46	0.22	-0.21	-1.08	-1.5
Clickbait	fullnetconc	**0.86**	**0.68**	**0.71**	0.56	0.29	0.04	0.01
	weNet	0.85	0.67	0.70	0.56	0.31	0.04	0.01
	lingNet	0.82	0.56	0.64	0.44	0.08	0.0	0.0
	fullNet	0.86	0.66	**0.71**	**0.59**	**0.37**	**0.06**	**0.01**
MDS Elect.	LogReg	0.76	0.74	0.52	0.34	0.16	0.05	0.03
	MLP1	0.75	0.71	0.5	0.33	**0.18**	**0.08**	**0.06**
	MLP4	0.73	0.71	0.47	0.24	-0.14	-0.78	-1.06
	mttri	**0.81**	**0.79**	**0.62**	**0.44**	0.15	-0.35	-0.58
MDS DVD	LogReg	0.74	**0.74**	0.48	**0.28**	0.12	0.04	0.03
	MLP1	0.73	0.73	0.46	0.27	**0.13**	**0.05**	**0.04**
	MLP4	0.72	0.72	0.44	0.20	-0.16	-0.74	-0.98
	mttri	**0.75**	0.72	**0.51**	**0.28**	-0.12	-0.84	-1.17
MDS Books	LogReg	0.70	0.68	0.41	0.23	**0.10**	**0.02**	**0.01**
	MLP1	0.69	0.66	0.38	0.13	0.01	-0.02	-0.01
	MLP4	0.7	0.68	0.39	0.15	-0.17	-0.67	-0.86
	mttri	**0.74**	**0.71**	**0.48**	**0.25**	-0.16	-0.87	-1.21
MDS Kitchen	LogReg	0.78	0.77	0.56	0.37	0.18	0.06	0.03
	MLP1	0.76	0.75	0.53	0.34	0.16	**0.07**	**0.04**
	MLP4	0.76	0.76	0.52	0.31	0.01	-0.48	-0.68
	mttri	**0.82**	**0.83**	**0.64**	**0.49**	**0.23**	-0.19	-0.38

Table 2: Comparison of accuracy, F1, and value for recalibrated models.

Task	Model	Acc	F1	Value for k=?				
				1	2	4	8	10
Hate Speech	Badj. et al	**0.82**	**0.63**	**0.64**	**0.51**	**0.36**	**0.27**	**0.22**
	Agr. et al.	0.73	0.62	0.46	0.21	0.0	0.0	0.0
Clickbait	fullnetconc	**0.86**	**0.68**	**0.71**	**0.61**	**0.49**	**0.37**	**0.33**
	weNet	0.85	0.67	0.70	0.6	0.47	0.36	**0.33**
	lingNet	0.82	0.56	0.64	0.5	0.35	0.17	0.11
	fullNet	**0.86**	0.66	**0.71**	0.6	**0.49**	**0.37**	**0.33**
MDS Elect.	LogReg	0.76	0.74	0.52	0.36	**0.23**	**0.12**	**0.1**
	MLP1	0.74	0.71	0.49	0.33	0.17	0.1	0.06
	MLP4	0.74	0.71	0.49	0.29	0.11	0.0	0.0
	mttri	**0.81**	**0.79**	**0.62**	**0.45**	0.19	0.11	0.0
MDS DVD	LogReg	0.74	**0.74**	0.48	**0.31**	**0.17**	**0.09**	**0.06**
	MLP1	0.73	0.73	0.46	0.28	0.15	0.04	0.02
	MLP4	0.72	0.72	0.44	0.23	0.06	0.0	0.0
	mttri	**0.75**	0.72	**0.51**	0.29	0.08	0.0	0.0
MDS Books	LogReg	0.70	0.68	0.41	0.23	**0.11**	**0.01**	0.0
	MLP1	0.7	0.66	0.39	0.2	0.0	0.0	0.0
	MLP4	0.69	0.68	0.37	0.09	0.0	0.0	0.0
	mttri	**0.74**	**0.71**	**0.48**	**0.26**	-0.01	0.0	0.0
MDS Kitchen	LogReg	0.78	0.77	0.56	0.41	**0.27**	**0.15**	**0.13**
	MLP1	0.76	0.75	0.51	0.34	0.2	0.1	0.01
	MLP4	0.75	0.76	0.50	0.30	0.12	0.0	0.0
	mttri	**0.82**	**0.83**	**0.64**	**0.49**	0.23	0.10	0.0

4.2 Results

Q1: Accuracy and F1-score are poor indicators of model value We first investigate whether standard performance metrics are sensible indicators of the model value, and how this depends on the magnitude of the cost factor k. Following the simplification in Section 3.2, we set $V_r = 0$ and $V_c = 1$, and use the threshold in Eq. 7 to decide whether to accept each prediction given a certain k.

Table 1 reports results in terms of accuracy, F1-score, and *value* for different values of $k \in [0, 10]$. As expected, the *value* of a model decreases substantially with the increase of the cost factor, with many models achieving *negative value* (i.e., it is better to simply ignore the model altogether) for larger values of k. We would like to stress that the cost factors we considered are fairly small and definitely realistic. For instance, setting $k = 4$ means that "being wrong is 4 times as bad" with respect to the advantage of being right. Many scenarios have values of k way more extreme (e.g., in medical decision support systems [60]). Notice that accuracy corresponds to the case where we do not reject any predictions (i.e. $k = 0$), a rather unrealistic scenario. Another major finding is that accuracy is a quite poor proxy of *value* even in relative terms. Boldface numbers indicate the best-performing model in terms of the different metrics. It is clear that the best-performing model is largely dependent on k, and that accuracy quickly becomes totally unreliable as a metric to identify the most appropriate model to employ. Replacing accuracy with F1-score doesn't change much. While we do observe substantially lower values for the unbalanced datasets (Hate Speech and Clickbait), the best-performing model is unchanged almost everywhere.

Q2: Cost-sensitive error is a poor indicator of model value in cost-sensitive settings The previous evaluation assumed equal cost for the different

types of error. This is however rarely the case in practical applications, where false negative errors (e.g., undiagnosed diseases) can be far more costly than false positive ones (i.e., false alarms). Section 3.4 shows how to adapt *value* to this cost-sensitive setting, and how to determine cost-sensitive thresholds that are specific for each predicted class. In the following, we evaluate the *value* of models in this cost-sensitive setting. We replace accuracy and F1, which are clearly inappropriate in this setting, with cost-sensitive error [20] a popular performance measure in the cost-sensitive learning literature. Cost-sensitive error is obtained by computing the weighted sum of errors, with the weights given by the corresponding cost, i.e. $(N_{fn}k_{fn} + N_{fp}k_{fp})/|\mathcal{D}|$, where we divide by $|\mathcal{D}|$ to remove the dependency on the size of the dataset. For simplicity, and consistently with common practice in the literature, we set $k_{fp} = 1$ and vary $k_{fn} \in [1, 10]$. Results are shown in Table 3. While cost-sensitive error identifies different best-performing models for different values of the cost, in only one case (MDS Kitchen) it consistently agrees with *value* across the spectrum of costs. What is worse, for large values of k_{fn} it often detects as best performing models that actually achieve *negative value*, making it a poor indicator of model *value*. The problem is not how it treats the costs of different errors, but in the fact that it does not assume a selective classifier and a corresponding cost-sensitive rejection threshold, which is the main practical contribution of our definition of *value*. This also implies that cost-sensitive learning [28], which aims at training classifiers to minimize cost-sensitive error, should be coupled with learning to reject mechanisms [31] to be fully effective in optimizing the *value* of the learned models.

Q3: Lack of calibration substantially affects model value The threshold in Eq. 7 assumes that models are perfectly calibrated, which is often far from

Table 3: Comparison of cost-sensitive error and value for $k = k_{fn}$ and $k_{fp} = 1$.

Task	Model	Cost-sensitive error					Value for k=?				
		$k=1$	$k=2$	$k=4$	$k=8$	$k=10$	1	2	4	8	10
Hate Speech	Badj. et al	**0.18**	**0.3**	0.53	1.01	1.25	**0.64**	**0.54**	**0.39**	**0.31**	**0.28**
	Agr. et al.	0.27	0.32	**0.43**	**0.64**	**0.75**	0.46	0.40	0.32	0.16	0.10
Clickbait	fullnetconc	**0.14**	**0.22**	**0.38**	**0.69**	**0.84**	**0.71**	0.61	0.37	0.13	**0.10**
	weNet	0.15	0.23	0.39	0.71	0.87	0.70	0.60	0.38	0.12	0.09
	lingNet	0.18	0.29	0.52	0.98	1.21	0.64	0.47	0.12	0.05	0.05
	fullNet	**0.14**	0.23	0.42	0.78	0.96	**0.71**	**0.63**	**0.45**	**0.15**	**0.10**
MDS Elect.	LogReg	0.24	0.41	0.74	1.41	1.75	0.52	0.44	**0.35**	**0.29**	**0.28**
	MLP1	0.26	0.44	0.79	1.5	1.85	0.5	0.41	0.34	0.28	0.27
	MLP4	0.25	0.42	0.74	1.4	1.73	0.47	0.33	0.09	-0.31	-0.49
	mttri	**0.19**	**0.33**	**0.61**	**1.16**	**1.44**	**0.62**	**0.49**	0.29	-0.08	-0.24
MDS DVD	LogReg	0.26	**0.39**	0.66	1.20	1.47	0.48	**0.37**	0.29	0.25	**0.25**
	MLP1	0.27	0.40	0.67	1.20	1.47	0.46	0.36	**0.3**	**0.26**	**0.25**
	MLP4	0.28	**0.39**	**0.62**	**1.07**	**1.30**	0.44	0.33	0.16	-0.09	-0.19
	mttri	**0.25**	0.41	0.74	1.41	1.74	**0.51**	0.35	0.07	-0.43	-0.66
MDS Books	LogReg	0.3	0.49	0.87	1.64	2.03	0.41	**0.33**	**0.27**	**0.22**	**0.22**
	MLP1	0.30	0.49	0.87	1.63	2.00	0.38	0.27	0.2	0.18	0.18
	MLP4	0.31	0.49	**0.83**	**1.52**	**1.87**	0.39	0.26	0.08	-0.18	-0.28
	mttri	**0.26**	**0.45**	**0.83**	1.60	1.99	**0.48**	0.32	0.02	-0.52	-0.79
MDS Kitchen	LogReg	0.22	0.34	0.6	1.11	1.36	0.56	0.47	0.36	0.31	0.29
	MLP1	0.24	0.37	0.64	1.17	1.44	0.53	0.43	0.34	0.29	0.28
	MLP4	0.25	0.39	0.66	1.22	1.5	0.52	0.42	0.26	0.03	-0.08
	mttri	**0.18**	**0.24**	**0.35**	**0.59**	**0.71**	**0.64**	**0.59**	**0.50**	**0.38**	**0.31**

Table 4: Comparison of acc., F1-score and value in OOD setting.

Task	Model	Acc	F1	Value for k=?				
				1	2	4	8	10
MDS Elect.	LogReg	0.76	0.74	0.52	0.34	0.16	0.05	0.03
	MLP1	0.74	0.71	0.5	0.33	0.18	0.08	**0.06**
	MLP4	0.74	0.71	0.47	0.24	-0.14	-0.78	-1.06
	mttri	0.81	0.79	0.62	0.44	0.15	-0.35	-0.58
	T5	0.78	0.76	0.57	0.35	-0.08	-0.94	-1.38
	SieBERT	**0.84**	**0.83**	**0.68**	**0.53**	0.22	-0.4	-0.70
	GPT-3	0.82	0.80	0.64	0.5	**0.32**	**0.13**	0.05
MDS DVD	LogReg	0.74	0.74	0.48	0.28	0.12	0.04	0.03
	MLP1	0.73	0.73	0.46	0.27	0.13	0.05	0.04
	MLP4	0.72	0.72	0.44	0.20	-0.16	-0.74	-0.98
	mttri	0.75	0.73	0.51	0.28	-0.12	-0.84	-1.17
	T5	0.79	0.79	0.58	0.37	-0.06	-0.9	-1.32
	SieBERT	**0.84**	**0.83**	**0.67**	0.51	0.19	-0.44	-0.75
	GPT-3	0.83	0.82	0.66	**0.53**	**0.37**	**0.16**	**0.09**
MDS Books	LogReg	0.7	0.68	0.41	0.23	0.10	0.02	**0.01**
	MLP1	0.7	0.66	0.38	0.13	0.01	-0.02	-0.01
	MLP4	0.69	0.68	0.39	0.15	-0.17	-0.67	-0.86
	mttri	0.74	0.71	0.48	0.25	-0.16	-0.87	-1.21
	T5	0.77	0.79	0.54	0.31	-0.15	-1.07	-1.52
	SieBERT	**0.83**	**0.83**	**0.65**	**0.48**	0.14	-0.55	-0.88
	GPT-3	0.81	0.81	0.61	0.46	**0.27**	**0.08**	**0.01**
MDS Kitchen	LogReg	0.78	0.77	0.56	0.37	0.18	0.06	0.03
	MLP1	0.76	0.75	0.53	0.34	0.16	0.07	0.04
	MLP4	0.75	0.76	0.52	0.31	0.01	-0.48	-0.68
	mttri	0.82	0.83	0.64	0.49	0.23	-0.19	-0.38
	T5	0.78	0.77	0.55	0.33	-0.11	-1.01	-1.45
	SieBERT	**0.86**	**0.86**	**0.73**	0.59	0.33	-0.19	-0.45
	GPT-3	0.85	0.85	0.71	**0.6**	**0.46**	**0.31**	**0.25**

being true for trained models, and deep learning models in particular [24]. In order to evaluate the role of calibration in determining the *value* of a model, we apply temperature scaling [24], a simple yet effective recalibration technique, to each model before applying the threshold (the resulting selector is shown in Figure 3(c)). Table 2 reports the results in exactly the same setting as Table 1, but using recalibrated models. Notice that accuracy and F1-score are unchanged, as temperature scaling affects the confidence in the prediction but not how classes are being ranked. In terms of *value*, however, we observe an overall improvement, quite substantial for larger values of k. The degenerate behavior of models with negative values is almost completely eliminated, with "useless" models receiving a *value* of zero. These results suggest that learning models should always be recalibrated before being incorporated into practical workflows. This does not mean that one can then resort to standard metrics to choose which model to employ. The best-performing model is still largely dependent on k. Notice that in the domain adaptation scenarios (MDS tasks), simple logistic regression (LogReg) consistently outperforms all other models for large values of k, as expected. Logistic regression is known to be a well-calibrated model per-se [37], and temperature scaling likely further improves this behavior, while more complex models struggle to achieve comparable calibration with simple recalibration strategies. The lively research area of calibration in ML and especially deep learning can provide useful solutions to this problem [58].

Q4: Operating in an OOD setting substantially affects model value The lack of calibration in ML models is known to be particularly harmful when the model operates in an OOD setting [64, 70], and the results on the domain adaptation tasks in Table 2 confirm this issue. To better understand the role of the OOD setting in determining the *value* of models, we focused on the MDS tasks and complemented the set of models presented in Table 2 with state-of-the-art transformer models, which should be less affected by the problem given the huge corpora they are trained. The employed transformer models are:

- Google's T5-base[11] [49] (12-layers, 768-hidden-state, 3072 feed-forward hidden-state, 12-heads, 220M parameters) fine-tuned on IMDB dataset[12] [42] for sentiment analysis task.
- SieBERT[13] [29]: a fine-tuned version of RoBERTa-large[14] model [41] (24-layer, 1024-hidden-state, 16-heads, 355M parameters) for sentiment analysis task that is fine-tuned and evaluated on 15 diverse text sources.
- GPT-3 [8]. Since it is producing human-like text for a given input, we fine-tuned it using the OpenAI API[15]. First, we prepared the MDS dataset for GPT-3; we cleaned sentences that have more than 2049 tokens, and renamed

[11] https://tinyurl.com/t5-base-finetuned-sentiment
[12] huggingface.co/datasets/imdb
[13] https://tinyurl.com/SieBERT-sentiment
[14] https://huggingface.co/roberta-large
[15] https://openai.com/api/

the text column as "*prompt*" and the ground truth column as "*completion*". Then, we used the OpenAI API to fine-tune GPT-3 separately on each of the 4 domains (DVD, books, electronics, and kitchen). We specified "number of classes" as 2 and the "positive class" as '1' so that the API tunes GPT-3 for binary sentiment analysis. Fine-tuning 4 models on the MDS dataset cost a total of $7.15. In order to test the fine-tuned models on different target domains, we specified the *prompt* in the format of "sentence + -> " because the API itself uses " ->" sign to teach GPT-3 that the sentiment for a *prompt* is (' ->') the *completion*. Thus, fine-tuned GPT-3 models produce either 0 or 1 for the given input. Testing each fine-tuned model on the other 3 domains (so, 12 cases in total) costs $43.89. We provide our source code on Github[16] to show every step of using GPT-3 in our experiments.

Table 4 reports the results of all models on the MDS tasks. As expected, large pre-trained language models tend to perform well across the board. This can be due to two reasons (besides the models being very powerful): (i) we know that very large models with very large train datasets are reasonably well calibrated (e.g. [33]), and (ii) when the training data is so large, fewer examples are out of distribution in terms of language. For example, GPT-3 [8] is trained on about 45TB of text data from various datasets, and the vocabulary of the MDS datasets is most likely already present in its training set.

Notice however that even for these models, accuracy is a poor proxy of value when k is large. Indeed, SieBERT slightly outperforms GPT-3 in terms of both accuracy and F1 in all tasks. However, the situation is reversed for large values of k, with SieBERT reaching negative values in most cases, most likely because of a poorer calibration with respect to GPT-3. Finally, simple linear models occasionally outperform these powerful (and very expensive to employ) large-language models for the largest values of k, again confirming the importance of value in determining the most appropriate model for the situation at hand.

5 Limitations and Conclusion

The takeaway from our experiments is that using accuracy-oriented metrics in hybrid decision-making systems is a risky proposition - and this is true even for models widely acknowledged as "leaders". We should constantly assess models over a range of cost factors, and at least for reasonable cost factors we expect based on the set of application use cases we are targeting. $k = 0$ (accuracy) is rarely a reasonable one. We also saw how applying models without thresholding can lead to a negative value, and that threshold tuning seems to perform better than calibration. We also hypothesize and have obtained some support for identifying complexity and out-of-distribution as factors that may lead to rapid model quality degradation for higher cost factors.

This being said, we see this work more as providing evidence of a problem and outlining the research needs: more studies (especially with large models and

[16] https://tinyurl.com/rethinking-value-of-ml-models

in vs out of distribution datasets) are required to validate the hypothesis and a deeper understanding of how calibration, confidence distribution, and size of validation set affect model value.

Acknowledgements The work of Andrea Passerini was partially supported by the PNRR project FAIR - Future AI Research (PE00000013), under the NRRP MUR program funded by the NextGenerationEU. The work of Burcu Sayin was partially supported by the project AI@Trento (FBK-Unitn).

References

1. Agrawal, S., Awekar, A.: Deep learning for detecting cyberbullying across multiple social media platforms. In: Pasi, G., Piwowarski, B., Azzopardi, L., Hanbury, A. (eds.) Advances in Information Retrieval. pp. 141–153. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-76941-7_11
2. Arango, A., Pérez, J., Poblete, B.: Hate speech detection is not as easy as you may think: A closer look at model validation. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 45–54. SIGIR'19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3331184.3331262, https://doi.org/10.1145/3331184.3331262
3. Badjatiya, P., Gupta, S., Gupta, M., Varma, V.: Deep learning for hate speech detection in tweets. In: Proceedings of the 26th International Conference on World Wide Web Companion. p. 759–760. WWW '17 Companion, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE (2017). https://doi.org/10.1145/3041021.3054223, https://doi.org/10.1145/3041021.3054223
4. Bahat, Y., Shakhnarovich, G.: Classification confidence estimation with test-time data-augmentation. ArXiv **abs/2006.16705** (2020). https://doi.org/10.48550/ARXIV.2006.16705
5. Balda, E., Behboodi, A., Mathar, R.: Adversarial examples in deep neural networks: An overview. In: Deep Learning: Algorithms and Applications. pp. 31–65 (01 2020). https://doi.org/10.1007/978-3-030-31760-7_2
6. Bendel, R., Higgins, S., Teberg, J., Pyke, D.: Comparison of skewness coefficient, coefficient of variation, and gini coefficient as inequality measures within populations. Oecologia **78**, 394–400 (03 1989). https://doi.org/10.1007/BF00379115
7. Bragg, J., Mausam, Weld, D.S.: Optimal testing for crowd workers. In: Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems. p. 966–974. AAMAS '16, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2016)
8. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D.: Language models are few-shot learners. ArXiv **abs/2005.14165** (2020). https://doi.org/10.48550/ARXIV.2005.14165, https://arxiv.org/abs/2005.14165

9. Bukowski, M., Kurek, J., Antoniuk, I., Jegorowa, A.: Decision confidence assessment in multi-class classification. Sensors **21**, 3834 (06 2021). https://doi.org/10.3390/s21113834
10. Callaghan, W., Goh, J., Mohareb, M., Lim, A., Law, E.: Mechanicalheart: A human-machine framework for the classification of phonocardiograms. In: CSCW'18. vol. 2, pp. 28:1–28:17 (2018). https://doi.org/10.1145/3274297
11. Casati, F., Noel, P., Yang, J.: On the value of ml models. In: Neurips workshop on Human Decisions (2021). https://doi.org/10.48550/ARXIV.2112.06775
12. Chai, X., Deng, L., Yang, Q., Ling, C.X.: Test-cost sensitive naive bayes classification. In: Fourth IEEE International Conference on Data Mining (ICDM'04). pp. 51–58 (2004). https://doi.org/10.1109/ICDM.2004.10092
13. Cheng, J., Bernstein, M.S.: Flock: Hybrid crowd-machine learning classifiers. In: Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (2015). https://doi.org/10.1145/2675133.2675214
14. Cordella, L., De Stefano, C., Tortorella, F., Vento, M.: A method for improving classification reliability of multilayer perceptrons. IEEE Transactions on Neural Networks **6**(5), 1140–1147 (1995). https://doi.org/10.1109/72.410358
15. De Stefano, C., Sansone, C., Vento, M.: To reject or not to reject: that is the question-an answer in case of neural classifiers. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) **30**(1), 84–94 (2000). https://doi.org/10.1109/5326.827457
16. Dellermann, D., Calma, A., Lipusch, N., Weber, T., Weigel, S., Ebel, P.A.: The future of human-ai collaboration: A taxonomy of design knowledge for hybrid intelligence systems. ArXiv **abs/2105.03354** (2019)
17. Dellermann, D., Ebel, P., Söllner, M., Leimeister, J.M.: Hybrid intelligence. Business & Information Systems Engineering **61**, 637–643 (10 2019). https://doi.org/10.1007/s12599-019-00595-2
18. Domingos, P.: Metacost: A general method for making classifiers cost-sensitive. In: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 155–164. KDD '99, Association for Computing Machinery, New York, NY, USA (1999). https://doi.org/10.1145/312129.312220, https://doi.org/10.1145/312129.312220
19. Elkan, C.: The foundations of cost-sensitive learning. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence - Volume 2. p. 973–978. IJCAI'01, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2001)
20. Elkan, C.: The foundations of cost-sensitive learning. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence. p. 973–978 (2001)
21. Fumera, G., Roli, F.: Support vector machines with embedded reject option. In: Proceedings of the First International Workshop on Pattern Recognition with Support Vector Machines. p. 68–82. SVM '02, Springer-Verlag, Berlin, Heidelberg (2002). https://doi.org/10.1007/3-540-45665-1_6
22. Gadiraju, U., Yang, J., Bozzon, A.: Clarity is a worthwhile quality: On the role of task clarity in microtask crowdsourcing. In: Proceedings of the 28th ACM Conference on Hypertext and Social Media. p. 5–14. HT '17, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3078714.3078715, https://doi.org/10.1145/3078714.3078715
23. Geifman, Y., El-Yaniv, R.: Selective classification for deep neural networks. In: Advances in Neural Information Processing Systems. vol. 30 (2017). https://doi.org/10.48550/ARXIV.1705.08500

24. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: Proceedings of the 34th International Conference on Machine Learning - Volume 70. p. 1321–1330. ICML'17, JMLR.org (2017). https://doi.org/10.48550/ARXIV.1706.04599
25. Han, L., Maddalena, E., Checco, A., Sarasua, C., Gadiraju, U., Roitero, K., Demartini, G.: Crowd worker strategies in relevance judgment tasks. In: Proceedings of the 13th International Conference on Web Search and Data Mining. p. 241–249. WSDM '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3336191.3371857, https://doi.org/10.1145/3336191.3371857
26. Han, L., Roitero, K., Gadiraju, U., Sarasua, C., Checco, A., Maddalena, E., Demartini, G.: All those wasted hours: On task abandonment in crowdsourcing. In: Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining. p. 321–329. WSDM '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3289600.3291035, https://doi.org/10.1145/3289600.3291035
27. Han, L., Roitero, K., Gadiraju, U., Sarasua, C., Checco, A., Maddalena, E., Demartini, G.: The impact of task abandonment in crowdsourcing. IEEE Transactions on Knowledge and Data Engineering **33**(5), 2266–2279 (2021). https://doi.org/10.1109/TKDE.2019.2948168
28. He, H., Ma, Y.: Imbalanced Learning: Foundations, Algorithms, and Applications. Wiley-IEEE Press (2013)
29. Heitmann, M., Siebert, C., Hartmann, J., Schamp, C.: More than a feeling: Benchmarks for sentiment analysis accuracy. In: Communication & Computational Methods eJournal (2020)
30. Hellman, M.E.: The nearest neighbor classification rule with a reject option. IEEE Transactions on Systems Science and Cybernetics **6**(3), 179–185 (1970). https://doi.org/10.1109/TSSC.1970.300339
31. Hendrickx, K., Perini, L., Van der Plas, D., Meert, W., Davis, J.: Machine learning with a reject option: A survey (2021)
32. Jiang, H., Kim, B., Guan, M.Y., Gupta, M.: To trust or not to trust a classifier. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. p. 5546–5557. NIPS'18, Curran Associates Inc., Red Hook, NY, USA (2018). https://doi.org/10.48550/ARXIV.1805.11783
33. Jiang, Z., Araki, J., Ding, H., Neubig, G.: How can we know when language models know? on the calibration of language models for question answering. Transactions of the Association for Computational Linguistics **9**, 962–977 (09 2021). https://doi.org/10.1162/tacl_a_00407
34. Kamar, E., Hacker, S., Horvitz, E.: Combining human and machine intelligence in large-scale crowdsourcing. In: AAMAS'12 - Volume 1. pp. 467–474 (2012)
35. Krivosheev, E., Casati, F., Benatallah, B.: Crowd-based multi-predicate screening of papers in literature reviews. In: Proceedings of the 2018 World Wide Web Conference. p. 55–64. WWW '18, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE (2018). https://doi.org/10.1145/3178876.3186036, https://doi.org/10.1145/3178876.3186036
36. Kull, M., Perello Nieto, M., Kängsepp, M., Silva Filho, T., Song, H., Flach, P.: Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with dirichlet calibration. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural

Information Processing Systems. vol. 32. Curran Associates, Inc. (2019), https://proceedings.neurips.cc/paper/2019/file/8ca01ea920679a0fe3728441494041b9-Paper.pdf
37. Kull, M., de Menezes e Silva Filho, T., Flach, P.A.: Beyond sigmoids: How to obtain well-calibrated probabilities from binary classifiers with beta calibration. Electronic Journal of Statistics **11**, 5052–5080 (2017)
38. Li, H.: Error rate analysis of labeling by crowdsourcing. In: International conference on machine learning (ICML2013), Workshop on machine learning meets crowdsourcing (2013)
39. Ling, C., Sheng, V.: Cost-sensitive learning and the class imbalance problem. In: Encyclopedia of Machine Learning (01 2010)
40. Liu, Q., Ihler, A.T., Steyvers, M.: Scoring workers in crowdsourcing: How many control questions are enough? In: Burges, C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K. (eds.) Advances in Neural Information Processing Systems. vol. 26. Curran Associates, Inc. (2013), https://proceedings.neurips.cc/paper/2013/file/cc1aa436277138f61cda703991069eaf-Paper.pdf
41. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. ArXiv **abs/1907.11692** (2019). https://doi.org/10.48550/ARXIV.1907.11692
42. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. pp. 142–150. Association for Computational Linguistics, Portland, Oregon, USA (Jun 2011), https://aclanthology.org/P11-1015
43. Nagar, Y., Malone, T.W.: Making business predictions by combining human and machine intelligence in prediction markets. In: International Conference on Interaction Sciences (2011)
44. Nagar, Y., Malone, T.W.: Improving predictions with hybrid markets. In: AAAI Fall Symposium: Machine Aggregation of Human Judgment (2012)
45. Ng, A.Y.: Feature selection, l1 vs. l2 regularization, and rotational invariance. In: Proceedings of the Twenty-First International Conference on Machine Learning. p. 78. ICML '04, Association for Computing Machinery, New York, NY, USA (2004). https://doi.org/10.1145/1015330.1015435, https://doi.org/10.1145/1015330.1015435
46. Nuñez, A.C.: Combining diverse forms of human and machine intelligence. In: PhD Thesis at Massachusetts Institute of Technology (2022)
47. Qarout, R., Checco, A., Bontcheva, K.: Investigating stability and reliability of crowdsourcing output. In: Proceedings of the 1st Workshop on Disentangling the Relation Between Crowdsourcing and Bias Management (CrowdBias 2018) co-located the 6th AAAI Conference on Human Computation and Crowdsourcing (HCOMP 2018) (07 2018)
48. Qiu, S., Gadiraju, U., Bozzon, A.: Improving worker engagement through conversational microtask crowdsourcing. In: Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems. p. 1–12. CHI '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3313831.3376403, https://doi.org/10.1145/3313831.3376403
49. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research **21**(1) (jun 2020). https://doi.org/10.48550/ARXIV.1910.10683

50. Raghu, M., Blumer, K., Corrado, G., Kleinberg, J.M., Obermeyer, Z., Mullainathan, S.: The algorithmic automation problem: Prediction, triage, and human effort. CoRR **abs/1903.12220** (2019)
51. Rodriguez, C., Daniel, F., Casati, F.: Crowd-based mining of reusable process model patterns. In: Business Process Management. pp. 51–66 (2014). https://doi.org/10.1007/978-3-319-10172-9_4
52. Ruder, S., Plank, B.: Strong baselines for neural semi-supervised learning under domain shift. In: The 56th Annual Meeting of the Association for Computational Linguistics (ACL 2018). pp. 1044–1054 (01 2018). https://doi.org/10.18653/v1/P18-1096
53. Sayin, B., Krivosheev, E., Passerini, J.Y.A., Casati, F.: A review and experimental analysis of active learning over crowdsourced data. Artificial Intelligence Review **54**, 5283–5305 (2021). https://doi.org/10.1007/s10462-021-10021-3
54. Sayin, B., Krivosheev, E., Ramírez, J., Casati, F., Taran, E., Malanina, V., Yang, J.: Crowd-powered hybrid classification services: Calibration is all you need. In: 2021 IEEE International Conference on Web Services (ICWS). pp. 42–50 (2021). https://doi.org/10.1109/ICWS53863.2021.00019
55. Sayin, B., Yang, J., Passerini, A., Casati, F.: The science of rejection: A research area for human computation. In: The 9th AAAI Conference on Human Computation and Crowdsourcing. HCOMP 2021, AAAI Press (2021). https://doi.org/10.48550/ARXIV.2111.06736
56. Shannon, C.E.: A mathematical theory of communication. The Bell System Technical Journal **27**(3), 379–423 (1948). https://doi.org/10.1002/j.1538-7305.1948.tb01338.x
57. Sheng, V.S., Ling, C.X.: Thresholding for making classifiers cost-sensitive. In: Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1. p. 476–481. AAAI'06, AAAI Press (2006)
58. de Menezes e Silva Filho, T., Song, H., Perelló-Nieto, M., Santos-Rodríguez, R., Kull, M., Flach, P.A.: Classifier calibration: How to assess and improve predicted class probabilities: a survey. CoRR **abs/2112.10327** (2021)
59. Suri, M.: PiCkLe at SemEval-2022 task 4: Boosting pre-trained language models with task specific metadata and cost sensitive learning. In: Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022). pp. 464–472. Association for Computational Linguistics, Seattle, United States (Jul 2022). https://doi.org/10.18653/v1/2022.semeval-1.63, https://aclanthology.org/2022.semeval-1.63
60. Sutton, R., Pincock, D., Baumgart, D., Sadowski, D., Fedorak, R., Kroeker, K.: An overview of clinical decision support systems: benefits, risks, and strategies for success. npj Digital Medicine **3** (2020)
61. Teerapittayanon, S., McDanel, B., Kung, H.T.: Branchynet: Fast inference via early exiting from deep neural networks. ArXiv **abs/1709.01686** (2017). https://doi.org/10.48550/ARXIV.1709.01686, https://arxiv.org/abs/1709.01686
62. Thai-Nghe, N., Gantner, Z., Schmidt-Thieme, L.: Cost-sensitive learning methods for imbalanced data. In: The 2010 International Joint Conference on Neural Networks (IJCNN). pp. 1–8 (2010). https://doi.org/10.1109/IJCNN.2010.5596486
63. Ting, K.M.: Inducing cost-sensitive trees via instance weighting. In: Proceedings of the Second European Symposium on Principles of Data Mining and Knowledge Discovery. p. 139–147. PKDD '98, Springer-Verlag, Berlin, Heidelberg (1998). https://doi.org/10.1007/BFb0094814

64. Tomani, C., Buettner, F.: Towards trustworthy predictions from deep neural networks with fast adversarial calibration. In: AAAI Conference on Artificial Intelligence (2019)
65. Tu, C.Y., Lin, H.T.: Cost learning network for imbalanced classification. In: 2020 International Conference on Technologies and Applications of Artificial Intelligence (TAAI). pp. 47–51 (2020). https://doi.org/10.1109/TAAI51410.2020.00017
66. Waseem, Z., Hovy, D.: Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In: Proceedings of the NAACL Student Research Workshop. pp. 88–93. Association for Computational Linguistics, San Diego, California (Jun 2016). https://doi.org/10.18653/v1/N16-2013, https://aclanthology.org/N16-2013
67. Whitehill, J., Wu, T.f., Bergsma, J., Movellan, J., Ruvolo, P.: Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C., Culotta, A. (eds.) Advances in Neural Information Processing Systems. vol. 22. Curran Associates, Inc. (2009), https://proceedings.neurips.cc/paper/2009/file/f899139df5e1059396431415e770c6dd-Paper.pdf
68. Wilder, B., Horvitz, E., Kamar, E.: Learning to complement humans. In: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence. IJCAI'20 (2021). https://doi.org/10.48550/ARXIV.2005.00582
69. Wu, M.H., Quinn, A.J.: Confusing the crowd: Task instruction quality on amazon mechanical turk. In: AAAI Conference on Human Computation & Crowdsourcing (2017)
70. Wu, Y., Zeng, Z., He, K., Mou, Y., Wang, P., Xu, W.: Distribution calibration for out-of-domain detection with Bayesian approximation. In: Proceedings of the 29th International Conference on Computational Linguistics. pp. 608–615. International Committee on Computational Linguistics, Gyeongju, Republic of Korea (Oct 2022), https://aclanthology.org/2022.coling-1.50
71. Yang, J., Redi, J., Demartini, G., Bozzon, A.: Modeling task complexity in crowdsourcing. In: AAAI Conference on Human Computation & Crowdsourcing (2016)
72. Zadrozny, B., Langford, J., Abe, N.: Cost-sensitive learning by cost-proportionate example weighting. In: Proceedings of the Third IEEE International Conference on Data Mining. p. 435. ICDM '03, IEEE Computer Society, USA (2003). https://doi.org/10.1109/ICDM.2003.1250950
73. Zhou, D., Basu, S., Mao, Y., Platt, J.: Learning from the wisdom of crowds by minimax entropy. In: Pereira, F., Burges, C., Bottou, L., Weinberger, K. (eds.) Advances in Neural Information Processing Systems. vol. 25. Curran Associates, Inc. (2012), https://proceedings.neurips.cc/paper/2012/file/46489c17893dfdcf028883202cefd6d1-Paper.pdf