Detecting and Mitigating Test-time Failure Risks via Model-agnostic Uncertainty Learning

Preethi Lahoti
Max Planck Institute for Informatics
plahoti@mpi-inf.mpg.de

Krishna P. Gummadi
Max Planck Institute for Software Systems
gummadi@mpi-sws.org

Gerhard Weikum
Max Planck Institute for Informatics
weikum@mpi-inf.mpg.de

***Abstract*—Reliably predicting potential failure risks of machine learning (ML) systems when deployed with production data is a crucial aspect of trustworthy AI. This paper introduces *Risk Advisor*, a novel post-hoc *meta-learner* for estimating failure risks and predictive uncertainties of *any already-trained* black-box classification model. In addition to providing a *risk score*, the *Risk Advisor* decomposes the uncertainty estimates into aleatoric and epistemic uncertainty components, thus giving informative insights into the sources of uncertainty inducing the failures. Consequently, *Risk Advisor* can distinguish between failures caused by data variability, data shifts and model limitations and advise on mitigation actions (e.g., collecting more data to counter data shift). Extensive experiments on various families of black-box classification models and on real-world and synthetic datasets covering common ML failure scenarios show that the *Risk Advisor* reliably predicts deployment-time failure risks in all the scenarios, and outperforms strong baselines.**

I. Introduction

Motivation and Problem: Machine learning (ML) systems have found wide adoption in mission-critical applications. Their success crucially hinges on the amount and quality of training data, and also on the assumption that the data distribution for the deployed system stays the same and is well covered by the training samples. However, this cannot be taken for granted. Saria and Subbaswamy [31] categorize limitations and failures of ML systems into several regimes, including data shifts (between training-time and deployment-time distributions), high data variability (such as overlapping class labels near decision boundaries) and model limitations (such as log-linear decision boundaries vs. neural ML). Trustworthy ML needs models and tools for detecting such failure risks and analyzing the underlying sources of uncertainty. Unfortunately, systems often fail silently without any warning, despite showing high confidence in their predictions [13, 20, 27].

This paper addresses the challenge of predicting, analyzing and mitigating failure risks for classifier systems. The goal is to provide the system with *uncertainty scores* for its predictions, so as to (a) reliably predict test-time inputs for which the system is likely to fail, and (b) detect the *type of uncertainty* that induces the risk, so that (c) appropriate *mitigation actions* can be pursued. Equipped with different kinds of uncertainty scores, a deployed system could improve its robustness in handling new data points that pose difficult situations. For instance, if an introspection component indicates that the ML system's output has a non-negligible likelihood of being erroneous, the system could abstain and defer the decision to a human expert (rather than risking adverse effect on human lives). When many production data points are out-of-distribution, collecting additional training samples and retraining the system would be a remedy. The challenge here is to determine which action is advised under which conditions. This is the problem addressed in this paper: determine the amount and type of uncertainty in deployment-time inputs, so as to decide if and which kind of mitigation is needed.

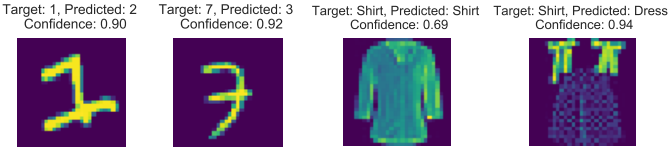


Fig. 1: Examples of ground-truth (target) and predicted labels where (a,b) a CNN fails despite high confidence (MNIST dataset [24]), and (c,d) a CNN assigns higher confidence to a misclassified sample than to a correct one (Fashion MNIST dataset [39])

State of the Art and its Limitations: The standard approach for deciding whether an ML system's predictions are trustworthy is based on confidence scores computed over predictive probabilities, such as max class probability (MCP) in neural ML [16] or distance from the decision boundary for SVM. However, predictive probabilities are not reliable estimates of a model's uncertainty [12, 13, 20, 27]. Figure 1(a,b) shows two examples where a CNN model misclassifies handwritten digits (from the MNIST benchmark) while giving high scores for its (self-) confidence. Even if the confidence scores are calibrated (e.g., [14, 30]), they may still not be trustworthy as the ordering of the confidence scores can itself be unreliable. This is because calibration methods are concerned with scaling of the scores, i.e., they perform monotonic transformations with respect to prediction scores, which do not alter the ranking of confident vs. uncertain example. Fig. 1(c,d) shows two examples where a CNN model gives higher score to a misclassified sample than to a correct one.

More importantly, confidence scores do not reflect what the model *does not know*. In Fig. 1(c,d), the Fashion MNIST dataset has many positive training examples of shirts similar to (c) while hardly any examples that resemble (d) – a case where the training distribution does not sufficiently reflect the test-time data. Yet, the CNN model makes a prediction with high confidence of 0.94 (see Fig. 1d). This limitation holds even for the state-of-the-art model *Trust Score* [20], which serves as a major baseline for this paper.

Moreover and most critically, *confidence scores* are "one-dimensional" and do not provide insight on which type of uncertainty is the problematic issue. Thus, confidence scores from prior works are limited in their support for identifying different types of appropriate mitigation.

A common line of work for uncertainty estimation builds on Bayesian methods [1, 5], or making specialized changes to the learning algorithm (e.g.,[6, 12, 23, 26, 36]). However, these are tightly coupled to the choice of the underlying classification model and thus involve making specialized modifications to the ML pipeline. Therefore, such techniques are unsuitable for dealing with a broad variety of black-box ML systems.

Proposed Approach: This paper presents *Risk Advisor*, a generic and versatile framework for reliably estimating failure risks of any already-trained black-box classification model. The *Risk Advisor* consists of a post-hoc *meta-learner* for uncertainty estimation that is separate from the underlying ML system, and can be incorporated without any code changes in the underlying ML pipeline. The *meta-learner* is model-agnostic: it can be applied to any family of black-box classifiers (e.g., deep neural networks, decision-trees, etc). Fig. 2 gives a schematic overview of our framework.

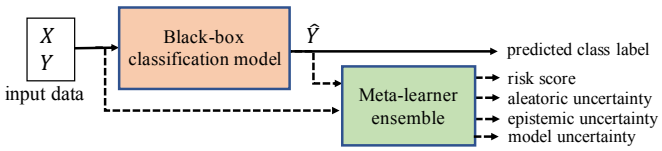


Fig. 2: Schematic overview of the Risk Advisor framework.

In addition to providing a *risk score* that is more reliable than those of prior works, the *Risk Advisor* provides a refined analysis of the underlying types of uncertainty inducing the risks. To this end, we make use of the information-theoretic notions of *model uncertainty*, *aleatoric uncertainty* and *epistemic uncertainty* [7, 18, 35]. These concepts are fairly old, but to the best of our knowledge, have not been considered for risk analysis of black-box ML systems. Our *Risk Advisor* quantifies each of the three risk types and thus enables judicious advise on risk mitigation action, depending on the type of uncertainty inducing the risks:

- *Aleatoric uncertainty* reflects the variability of data points and the resulting noise around the classifier's decision boundary. A high value indicates that it is inherently difficult to distinguish the output classes, and an appropriate mitigation then is to equip the deployed system with the option to *abstain* rather than forcing an output label. Fig. 1(a,b) is a case of high aleotoric uncertainty.
- *Epistemic uncertainty* captures systematic gaps in the training samples, like regions where training samples are sparse but have a substantial population of test points after deployment. This situation can only be countered by obtaining *more training data* for the underrepresented critical regions. Fig. 1(c,d) is a case of high epistemic uncertainty.
- *Model uncertainty* is an indicator that the black-box ML system uses models with insufficient learning capacity. In this situation, the proper action is to re-build the ML system with higher model capacity or a more expressive learning model, for example, a deep neural network instead of a log-linear model or Transformer instead of LSTM or CNN.

The proposed *meta-learner* for estimating the different types of uncertainty in the *Risk Advisor* framework is implemented as an ensemble of M stochastic gradient-boosted decision trees (E-SGBT). Each stochastic gradient boosted tree (SGBT) operates on the input-output pairs of training samples and an indicator variable stating whether the trained black-box ML system misclassified the training point. The *Risk Advisor's* analysis of uncertainty is based on the ensemble's ability to compute aleatoric and epistemic uncertainty. All of the uncertainty scores are computed on the training data, and also at deployment time for test data alone to identify slowly evolving risks.

Contributions: The state-of-the-art method to which we compare our approach is *Trust Score* [20], which also operates in a model-agnostic post-hoc way. Trust Score is based on the distance between a test point and its nearest neighbors in the training data. Its output is a single value, which does not provide guidance on identifying the type of risk.

This paper's novel contributions are as follows:

- We introduce the *Risk Advisor* framework, the first *model-agnostic* method to detect and mitigate deployment-time failure risks, requiring access only to the base classifier's training data and its predictions and coping with any kind of underlying Black-box ML model.
- The *Risk Advisor* is the first method that leverages the information-theoretic notions of *aleatoric* and *epistemic* uncertainty to distinguish between ML model failures caused by distribution shifts between training data and deployment data, inherent data variability, and model limitations.
- Experiments with synthetic and real-world datasets show that our approach successfully detects uncertainty and failure risks for many families of ML classifiers, including deep neural models, and does so better than prior baselines including the Trust Score method by Jiang et al. [20].
- We demonstrate the *Risk Advisor*'s practical utility by three kinds of risk mitigation: (i) selectively abstaining from making predictions under uncertainty (ii) detecting out-of-distribution test-examples (iii) countering risks due to data shift by collecting more training samples in a judicious way.

II. Related Work

The standard approach for predicting failure risks of ML systems is to rely on the system's native (self-) *confidence scores*. An implicit assumption is that that most uncertain data points lie near the decision boundary, and confidence increases when moving away from the boundary. While this is reasonable to capture *aleatoric* uncertainty, this kind of confidence score fails to capture *epistemic* and *model* uncertainty [12]. Techniques for confidence calibration [14, 30] are concerned with re-scaling the model's prediction scores to produce calibrated probabilities. Like all single-dimensional notions of confidence, this is insufficient to distinguish different types of uncertainty and resulting risks. In particular, there is no awareness of epistemic uncertainty due to data shifts [28].

Bayesian methods are a common approach to capture uncertainty in ML [1, 5]. Recently, a number of non-Bayesian specialized learning algorithms were proposed to approximate Bayesian methods. For instance, variational learning [17, 21], drop-out [12], and ensembles of deep neural networks [23]. However, these models tend to be computationally expensive (by increasing network size and model parameters), and are not always practically viable. Moreover, they require changes to the architecture and code of the underlying ML system.

The concepts of aleatoric and epistemic uncertainty are rooted in statistics and information theory [7, 18] ([19] is a recent overview). [35] has incorporated these measures into a Bayesian classifier with fuzzy preference modeling. [36] integrated the distinction between aleatoric and epistemic uncertainty into random-forest classifiers to enhance its robustness. Both of these works are focused on one specific ML model and do not work outside these design points, whereas Risk Advisor is model-agnostic and as such universally applicable. [36] is included in the baselines for our experimental comparisons.

Several post-hoc approaches were proposed for confidence scores of already trained classifiers. Schulam and Saria [34] proposed a post-hoc auditor to learn pointwise reliability scores. However, it is not model-agnostic as it relies on using gradients and the Hessian of the underlying ML model. Further, it does not differentiate between different types of uncertainty. Schelter et al. [32] proposed a *model-agnostic* validation approach to detect *data-related* errors at serving time. However, this work focuses on errors arising from data-processing issues, such as missing values or incorrectly entered values, and relies on programmatic specification of typical data errors.

The closest approach to ours is *Trust Score* [20], a model-agnostic method that can be applied post-hoc to any ML system. Trust Score measures the agreement between a classifier's predictions and the predictions of a modified nearest-neighbour classifier which accounts for density distribution. More precisely, the *trust score* for a new test-time data point is defined as the ratio between (a) the distance from the test sample to its nearest α-high density set with a *different* class and (b) the distance from the test sample to its nearest α-high density set with the *same* class. A crucial limitation of this approach is that it is highly sensitive to the choice of the distance metric for defining neighborhoods, and can degrade for high-dimensional data. Also, it does not provide any guidance on the type of uncertainty.

Classification with reject option [2] and selective abstention [9] are related problems, where the model can defer decisions (e.g., to to a human expert) when it has low confidence. However, these methods still rely on their own *confidence* scores to determine when to abstain, and thus share the limitations and pitfalls of a single-dimensional self-confidence. Similarly, the problem of detecting data shifts has been widely studied e.g., for detecting and countering covariate and label shift[33] and for anomaly detection [3, 37]. These methods address data shifts, but they do not consider failure risks arising from *aleatoric* and *model* uncertainty.

III. Risk Advisor Model

A. Basic Concepts

Black-box Classifier's Task: We are given a training dataset $\mathcal{D} = \{(x_i, y_i) \cdots (x_n, y_n)\} \subset \mathcal{X} \times \mathcal{Y}$ drawn from an unknown data generating distribution $\mathcal{P} \sim \mathcal{X} \times \mathcal{Y}$. The goal of the *black-box classifier* is to learn a hypothesis h that minimizes the expected empirical risk over observed training distribution $\mathcal{D}$.

$$h^* = \arg\min_h \mathbb{E}_{(x,y)\in\mathcal{D}} \ell(h(x), y) \tag{1}$$

where $\ell(\cdot)$ is classification loss function (e.g., cross-entropy between predicted and ground-truth labels), and $\hat{y} = h(x)$ is the corresponding predicted class label.

Black-box Classifier's Uncertainty: The degree of uncertainty in a prediction can be measured by the Shannon entropy over the outcomes for any given test point. Higher entropy corresponds to higher uncertainty.

$$H[Y|X] = -\sum_{y\in\mathcal{Y}} P(y|x, \mathcal{D}) \log_2 P(y|x, \mathcal{D}) \tag{2}$$

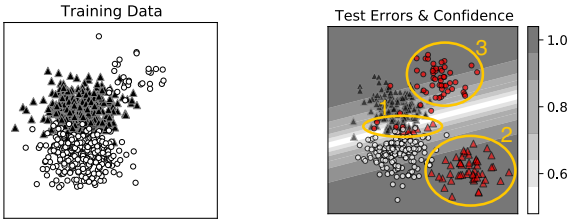
The overall uncertainty corresponding to the predictive task denoted as $H[Y|X]$ encompasses uncertainty due to aleatoric, epistemic and model uncertainty [7, 18, 35].

B. Mapping Failure Scenarios to Uncertainties

Next, we give a brief introduction to the types of uncertainties – aleatoric, epistemic and model uncertainty – in a predictive task, and draw a connection between predictive uncertainties and common sources of failures in ML system.

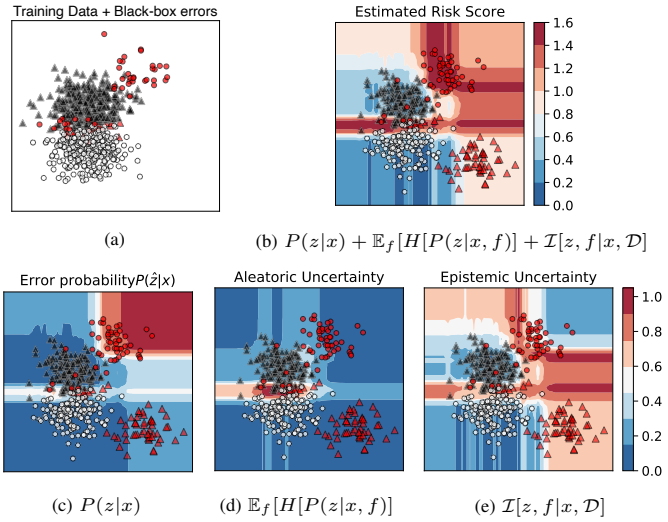
Example: These different kinds of uncertainty are illustrated via a synthetic example in Fig. 3. We will use this as running example to motivate the proposed approach. Consider the classification task dataset in Fig. 3. The position on x-axis and y-axis represents input features. The markers (black triangles and white circles) represent binary class labels. A linear SVM classifier, for example, would learn a decision boundary that best discriminates the two classes as shown in Fig. 3b. The test-errors made by the model are highlighted in red.

Firstly, in many predictive tasks Y can rarely be estimated deterministically from X due to inherent stochasticity in the dataset, a.k.a *aleatoric* uncertainty. For instance, errors arising due to inherent data variability and noise, marked as Region 1 in Fig. 3b). Such errors are inherently *irreducible* (unless additional features are collected). Additionally, there is uncertainty arising due to "lack of knowledge" about the true data generating process. For instance, consider the test errors caused by shifts in the data distribution, marked as Region 2. Such errors due to *epistemic* uncertainty can in principle be mitigated by collecting additional training data and retraining the model. Further, ML models have additional uncertainty in estimating the true model parameters given limited training data. For instance, consider *systematic* errors arising due to fitting a linear model to non-linear data, marked as Region 3. Errors due to *model* uncertainty can in principle be addressed (e.g., by training a model from a different model class).



(a) (b)

Fig. 3: *Example*: (a) Training data for classification task (b) learned decision boundary of an SVM classifier and different types of test-time errors, e.g., due to (1) data variability and noise (2) data shift, and (3) model limitations.



(a) (b) $P(z|x) + \mathbb{E}_f[H[P(z|x,f)] + \mathcal{I}[z,f|x,\mathcal{D}]$

(c) $P(z|x)$ (d) $\mathbb{E}_f[H[P(z|x,f)]$ (e) $\mathcal{I}[z,f|x,\mathcal{D}]$

Fig. 4: Meta-learner: (a) Training input to *meta-learner* (b) *meta-learner's* estimated overall *risk score* (c, d, e) decomposition of the overall risk score into its various constituting components, i.e., (c) model, (d) aleatoric and (e) epistemic uncertainty, that capture errors due to (c) model limitations, (d) data variability and noise, and (e) data shift, respectively.

C. *Design Rationale*

We draw inspiration from *Fano's Inequality* [4, 10], a classic information-theoretic inequality which when viewed from a ML perspective draws a connection between predictive uncertainty $H[Y|X]$, uncertainty in error prediction $H[Z|X]$, and probability of error $P(Z|X)$ of a Bayes optimal classifier, where Z is a random variable indicating prediction error $Z := \mathbb{I}(Y \neq \hat{Y})$.

Fano's Inequality [4, 10]: *Consider random variables X and Y, where Y is related to X by the joint distribution $P(x, y)$. Let $\hat{Y} = h(X)$ be an estimate of Y, with the random variable Z representing an occurrence of error, i.e., $Z := \mathbb{I}(Y \neq \hat{Y})$. Fano's inequality states that*

$$H[Y|X] \leq H[Z|X] + P(Z|X) \cdot \log_2(|\mathcal{Y}| - 1) \quad (3)$$

where $|\mathcal{Y}|$ is the number of classes, H is Shannon entropy, and $P(Z|X)$ is probability of error.

Key Idea: The conditional entropy $H[Z|X]$ and the error probability $P(Z|X)$ in Eq. 3 are not known, but we can approximate them by computing empirical estimates of conditional entropy $H_f[Z|X]$ and error probability $P_f(Z|X)$ of a separate *meta-learner* $f : X \rightarrow Z$ whose goal is to predict errors Z made by the underlying black-box classifier h with respect to the original classification task.

Given such a meta-learner f, we argue that a black-box model's classification errors on unseen data, which relate to the uncertainty $H[Y|X]$, can be estimated by combining f's predicted probability of error $P_f(Z|X)$ and f's own uncertainty corresponding to predicting errors $H_f[Z|X]$.

Example: Let us revisit the synthetic example of Fig. 3, looking at it from a meta-learner's perspective. Fig. 4 shows different perspectives on this setting.

Fig. 4a visualizes the input to the meta-learner, which consists of training datapoints X and the black-box model's training errors Z (highlighted in red). Observe that errors due to *model limitations* (top right red points) appear as systematic errors in the input space, and are *predictable*. We argue that by training a *meta-learner* to predict black-box classification model's errors, we can capture these systematic errors due to model limitations with meta-learner's predicted *error probabilities* $P_f(Z|X)$, as shown in Fig. 4c.

Further, recall that both aleatoric and epistemic uncertainties are related to the underlying training data. We posit that the meta-learner, which is trained on the same data samples as the black-box classifier, inherits these data-induced uncertainties, and this is reflected in the meta-learner's *aleatoric* and *epistemic* uncertainties, as shown in 4d and Fig. 4e.

The intuition is as follows. Consider the region near the decision boundary in Fig. 4a. As the meta-learner sees both *failure* and *success* cases of the black-box classifier in this region, the meta-learner, too, has *aleatoric uncertainty* in this region of inherent noise. Similarly, consider the test points situated far away from the training data. The meta-learner would also have significant epistemic uncertainty in its error prediction, as it has not seen any training data in this region. Thus, by estimating the meta-learner's *own* aleatoric and epistemic uncertainty, we can indirectly capture the black-box classifier's aleatoric and epistemic uncertainty, as shown in Fig. 4d and Fig. 4e, respectively. In our experiments, we will present empirical evidence of these insights.

Putting these three insights together, we propose the combined notion of *risk score*, as shown in Fig. 4b. The estimated risk score (background color) is high in the regions of actual test-time errors.

In the following, Subsection III-D formalizes the meta-learner's task, and presents our proposed meta-learner ensemble for the Risk Advisor. Subsection III-E discusses how to refine the overall uncertainty into informative components for different kinds of uncertainty, and compute overall *risk score*.

D. *Meta-learner Ensemble*

Meta-learner's Task: Given input training samples $x \in X$, predicted class labels $\hat{y} := h(x)$ of a fully trained *black-box classifier* h, and a random variable $Z := \mathbb{I}(Y \neq \hat{Y})$ indicating errors of the *black-box classifier* h with respect to the original classification task. Our goal is to learn an meta-learner $f : X \rightarrow Z$ trained to predict errors of the *black-box classifier* with respect to the original task given by

$$f = \arg\min_{f \in \mathcal{F}} \mathbb{E}_{(x,z)\in\mathcal{D}} \ell(f(x), z) \quad (4)$$

where z is a random variable indicating errors of the base classifier predictor given by $z = \mathbb{I}(y \neq \hat{y})$, ℓ is a classification loss function. Given a newly seen test point x^*, the *meta-learner's* predicted probability of error is given by $P(z|f, x^*)$.

However, the probability of error $P(z|f, x^*)$ estimated by a single meta-learner f can be biased due to its own uncertainty in the model parameters $P(f|\mathcal{D})$. Next, we show how we can obtain a reliable estimate of the black-box model's error probability by training an ensemble of M independent stochastic gradient boosted trees $\mathcal{F} = \{P(z|x^*, f^m)\}_{m=1}^M$, and computing their expectation.

Ensemble of Stochastic Gradient Boosted Trees (E-SGBT):

We consider an ensemble of M independent models $\mathcal{F} = \{f^m\}_{m=1}^M$ such that each of the individual models f^m is a stochastic gradient boosted tree (SGBT) [11]. Note that the proposed *E-SGBT* is an ensemble of ensembles, i.e., each of the M SGBT's in the ensemble is itself an ensemble of T weak learners trained iteratively via bootstrap aggregation. To ensure minimum correlation between the M individual models in our ensemble, we introduce randomization in two ways. First, each of the SGBTs in the ensemble is initialized with a different random seed. Second, each of the individual SGBTs is itself an ensemble of T weak learners trained iteratively via bootstrap aggregation. Specifically, for each SGBT in the *E-SGBT* ensemble, at each iteration, a subsample of training data of size $\tilde{N} < N$ is drawn at random, without replacement, from the full training dataset. The fraction $\frac{\tilde{N}}{N}$ is called the sample rate. The smaller the sample rate, the higher the difference between successive iterations of the weak learners, thereby introducing randomness into the learning process.

Given M error probability estimates $\{P(z|x, f^m)\}_{m=1}^M$ by each of the models in the ensemble, an estimate of the probability of error $P(z|x, \mathcal{D})$ can be computed by taking the expectation over all the models in the ensemble:

$$P(z|x, \mathcal{D}) := \mathbb{E}_{f \in \mathcal{F}}[P(z|x, f, \mathcal{D})] \approx \frac{1}{M} \sum_{m=1}^{M} P(z|x, f^m, \mathcal{D}) \tag{5}$$

The total uncertainty in the error prediction $H[P(z|x, \mathcal{D})]$ can be computed as the Shannon entropy corresponding to the estimated probability of error

$$H[P(z|x, \mathcal{D})] = -\sum_{z \in \mathcal{Z}} P(z|x, \mathcal{D}) \log_2 P(z|x, \mathcal{D}) \tag{6}$$

E. Identifying Sources of Uncertainty

To distinguish between different sources of uncertainty – data variariability/noise vs. data shifts between training and deployment data – we compute estimates of the *aleatoric* and *epistemic* uncertainty given an ensemble of M independent stochastic gradient boosted trees $\mathcal{F} = \{f^m\}_{m=1}^M$. This approach was originally developed in the context of neural networks [6], but the idea is more general and has recently been applied using ensembles of gradient boosted trees and random forests [26, 36].

Decomposing Aleatoric and Epistemic Uncertainty: The main idea is that in the case of data points with epistemic uncertainty (e.g., out-of-distribution points), the M independent models in the ensemble given $\mathcal{F} := \{f^m\}_{m=1}^M$ are likely to yield a diverse set of predictions (i.e., different output labels) for similar inputs. In contrast, for data points with low epistemic uncertainty (e.g., in-distribution points in dense regions), they are likely to agree in their predictions. Hence, by fixing f, the *epistemic* uncertainty can be removed, and the *aleatoric* uncertainty can be computed by taking the expectation over all models $f \in \mathcal{F}$.

$$\mathbb{E}_{p(f|\mathcal{D})} H[P(z|x, f)] = \int_{\mathcal{F}} P(f|D) \cdot H[P(z|f, x)] df \tag{7}$$

Aleatoric Uncertainty: Given M predicted probability estimates $\{P(z|x, f^m)\}_{m=1}^M$ for each of the models in the ensemble, an estimate of *aleatoric uncertainty* in Eq. 7 can be empirically approximated by averaging over individual models $f^m \in \mathcal{F}$ in our *E-SGBT* ensemble.

$$\mathbb{E}_{f \in \mathcal{F}}[H[P(z|x, f)]] \approx \frac{1}{M} \sum_{m=1}^{M} H[P(z|x, f^m)] \tag{8}$$

Epistemic uncertainty: Finally, *epistemic uncertainty* can be computed as the difference between *total uncertainty* and *aleatoric uncertainty*.

$$\underbrace{\mathcal{I}[z, f|x, \mathcal{D}]}_{\text{Epistemic Uncertainty}} = \underbrace{H[P(z|x, \mathcal{D})]}_{\text{Total Uncertainty}} - \underbrace{\mathbb{E}_{f \in \mathcal{F}}[H[P(z|x, f)]]}_{\text{Aleatoric Uncertainty}} \tag{9}$$

where *total uncertainty* is the entropy corresponding to the estimated probability of error $P(z|x, \mathcal{D})$ given in Eq. 6.

Risk Score: Putting it all together, our proposed *Risk Score*, which captures black-box model errors arising due to all sources of uncertainty, can be computed as the sum of (i) predicted probability of error assigned by the meta-learner, i.e., model uncertainty, (ii) epistemic uncertainty and (iii) aleatoric uncertainty.

$$\begin{aligned} \text{Risk Score} &:= \underbrace{P(z|x, \mathcal{D})}_{\text{Error probability}} + \underbrace{H[P(z|x)]}_{\text{Total uncertainty}} \\ &= \underbrace{P(z|x, \mathcal{D})}_{\text{Model Uncertainty}} + \underbrace{\mathcal{I}[z, f|x, \mathcal{D}]}_{\text{Epistemic uncertainty}} + \underbrace{\mathbb{E}_f[H[P(z|x, f)]}_{\text{Aleatoric uncertainty}} \end{aligned} \tag{10}$$

Note that this *risk score* is neither a probability nor an entropy measure, but it proves to be a very useful indicator for failure risks in our experiments. One could consider a weighted sum of each of the components to account for associated *risk costs* for each type of error. For instance, if a system designer had expert knowledge that errors due to distribution shift (i.e., epistemic uncertainty) are more harmful, she could assign more weight to the *epistemic uncertainty* component. In our experiments we assign equal weights.

Inference: The meta-learner is trained on the underlying base-classifier's training data. Given a newly seen test point x^* at deployment-time, the *Risk Advisor* computes predicted error probabilities for each of the M models in the E-SGBT ensemble $\{P(z|x^*, f^m)\}_{m=1}^M$. These values are fed into the *Risk Advisor's* estimated *error probability* in Eq. 5, aleatoric uncertainty in Eq. 8, epistemic uncertainty in Eq. 9 and risk score in Eq. 10. Note that at deployment-time, we only expect the unseen data point x^*, and the trained meta-learner.

IV. Synthetic-Data Experiments

In this section, we evaluate *Risk Advisor's* ability to detect the sources of uncertainty inducing the failure risk. To this end, we systematically generate synthetic datasets covering a variety of ML failure scenarios including errors due (i) black-box classifier's model limitations (e.g., applying a linear model to non-linear decision boundary) (ii) data shift and (iii) inherent data variability and noise. We then evaluate if the *Risk Advisor's* estimates for *model*, *epistemic*, and *aleatoric* uncertainty can correctly capture the corresponding test-time errors made by the black-box classification model.

Errors due to Black-box Classifier's Model Limitations: In order to simulate this scenario, we construct a classification dataset with a non-linear decision boundary, i.e., two concentric circles [29]. We then fit a misspecified classification model to the task, i.e., a logistic regression classifier with a (log-)linear decision boundary as shown in Fig. 5. The contour plot in Fig. 5a visualizes the training data and the learned decision boundary. Fig. 5b visualizes the test data. Test-set errors made by the black-box model are highlighted in red.

The contour plot in Fig. 5c visualizes the *Risk Advisor's* predicted *Error probability* ($P(\hat{z}|x)$). Ideally, we would expect the *Risk Advisor* to assign a higher error score for regions of the input space where the black-box classifier makes errors due to its model limitations. We clearly see this trend: the *Risk Advisor* correctly identifies the regions where the black-box classifier is likely to make errors due to its incorrect linear decision boundary. This is especially remarkable given that the *Risk Advisor* has no knowledge of the model family of the underlying black-box model (e.g., whether it is log-linear model or a neural network). In spite of having no information about the underlying model (other than its predictions), the *Risk Advisor* is able to correctly capture the *model uncertainty*.

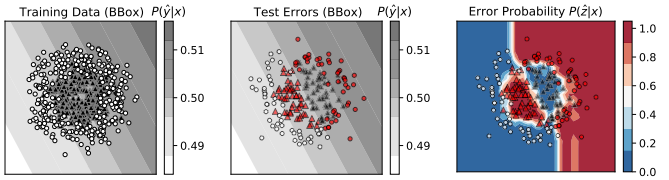


Fig. 5: Errors from model limitations: *Risk Advisor's* estimated error probability $P(z|x)$ correctly identifies errors due to model limitation.

Errors due to Distribution Shift: In order to simulate a distribution shift scenario, we draw points from a mixture of two Gaussians. For the training points we set the mixture coefficient for one of the Gaussians to zero; for the test points both mixture components are active. This way, we are able to construct a dataset containing out-of-distribution test points as shown in Fig. 6. Fig. 6a visualizes the training data and the decision boundary learned by a 2-layer feed-forward neural network (NN). Fig. 6b visualizes the test data. Test errors of the NN are highlighted in red. Observe that the NN *misclassifies* out-of-distribution test points while (incorrectly) reporting high confidence. The contour plot in Fig. 6c visualizes *Risk Advisor's* estimated *epistemic* uncertainty.

Ideally, we would like to see that the *epistemic* uncertainty increases as we move towards the sparse regions of the training data, and that it is high for out-of-distribution regions. Despite some noise, we clearly see this trend: regions of low epistemic uncertainty (i.e., dark-blue regions) coincide with the dense in-distribution test points. *Epistemic* uncertainty increases as we move towards sparse regions, and the values are especially high for out-of-distribution regions (bottom right in Fig. 6c).

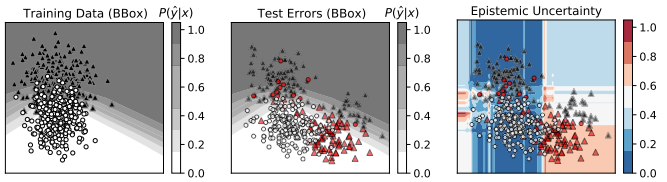


Fig. 6: Errors from distribution shift: Risk-Advisor's epistemic uncertainty correctly identifies test points far away from training distribution.

Errors due to Data Variability and Noise: To simulate a dataset with inherent noise, we draw points from the classic two-moons dataset [29], and add Gaussian noise with standard deviation 0.5 to the dataset as shown in Fig. 7. Fig. 7a visualizes the training data and the decision boundary learned by a 2-layer feed-forward neural network (NN). Fig. 7b visualizes the test data. Test-errors are highlighted in red.

The contour plot in Fig. 7c visualizes estimated *aleatoric* uncertainty. Ideally, we would expect that *aleatoric* uncertainty is high for the regions with large class overlap. We clearly see this trend: the estimated *aleatoric* uncertainty is high for the test points near the decision boundary, with high class overlap.

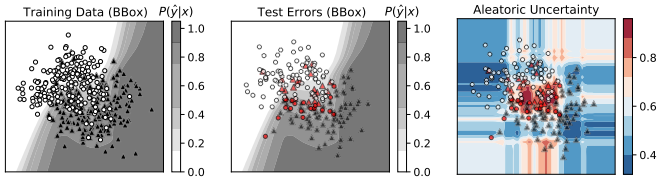


Fig. 7: Errors from data variability and noise: Risk Advisor's aleatoric uncertainty correctly identifies test points in the regions with class overlap.

V. Real-World-Data Experiments

In this section, we evaluate the performance of *Risk Advisor* by performing extensive experiments on 9 real-world datasets and on 6 families of black-box classification models. First, we evaluate the *Risk Advisor's* ability to *predict failure risks* at deployment time (Subsection V-B). Next, we investigate its performance on a variety of applications for *risk mitigation*, including (i) selectively abstaining under uncertainty (Subsection V-C) (ii) detecting out-of-distribution test examples (Subsection V-D) and (iii) mitigating risk by judiciously collecting additional samples for re-training the system (Subsection V-E).

A. *Experimental Setup*

Datasets: We perform our evaluation on the following small and large benchmark classification datasets covering a variety of common ML failure scenarios:

High-dimensional Image Datasets:

- *CIFAR 10:* The CIFAR-10 dataset [22] consists of 60K color images in 10 classes, including blurred and noisy images, which are specially prone to model failures.
- *MNIST:* The MNIST dataset [24] consists of 60K grayscale images of handwritten digits in 10 classes. Due to the variability in writing style, certain images are prone to misclassification.
- *Fashion MNIST:* The fashion MNIST dataset [39] consists of 60K images of clothing and accessories in 10 classes, including images with rare and unusual product designs, which can be prone to errors.

Mission-critical Fairness Datasets:

- *Census Income:* Recent work in ML fairness has shown that models often make more errors for underrepresented groups in training data. To simulate this setting, we consider the Adult dataset [8], a benchmark dataset in fairness literature, consisting of 49K user records. The dataset contains underrepresented groups (e.g., Female).
- *Law School:* Similarly, we use the LSAC dataset [38] consisting of 28K law school admission records. The classification task is to predict whether a candidate would pass the bar exam. The dataset contains underrepresented groups (e.g., "Black").

Distribution shift, unseen demographics/regions/domain:

- *Census Income (Male $\rightarrow$ Male, Female):* To simulate distribution shift, we take the aforementioned *Census Income* dataset, and exclude *female* points from the training set. Our test set consists of both Male and Female points.
- *Law School (White $\rightarrow$ White, Black):* Similarly, we take the aforementioned *Law School* dataset, and exclude user records from the *Black* demographic group from the training set. The test set consists of both White and Black points.
- *Heart Disease:* A common ML failure scenario is when a ML model is applied to a new geographic region. To simulate this scenario we combine four different heart disease datasets available in the UCI repository [8] by using a subset of features overlapping between them. We use the US Cleveland heart disease dataset as our training dataset, and use it to predict heart disease on a UK statlog dataset, Hungarian (HU) and Switzerland (CH) heart disease dataset.
- *Wine Quality:* Another failure scenario is when a trained model is applied to an application domain for which it has inadequate or bad training data. To simulate this scenario, we train models on white wine, and apply it to predict quality of red wine in UCI wine dataset [8]. The classification task is to predict if the wine quality is ≥ 6.

Black-box Classification Models: To demonstrate the versatility of *Risk Advisor*, we evaluate it on classifiers from 6 different families, including deep neural models such as ResNet50 and CNN for the high dimensional image dataset, and classic ML algorithms such as SVM, Random Forests, Multi-layer Perceptron, and logistic regression for tabular datasets. Following are the implementation details:

- ResNet 50: The 50-layer deep residual network architecture [15] trained with batch size of 128 for 100 epochs.
- CNN: A convolutional neural net with 2 convolutional layers with 32, 64 hidden units, max pooling, and ReLu activations, trained with batch size 128 for 10 epochs.
- MLP: Multi-layer perceptron with 2 hidden layers with $32, 16$ hidden units, batch size 64, and ReLu activations.
- SVM: support vector machines with RBF kernel and Platt scaling [30] to produce probability estimates.
- RF: a random forest with 1000 decision trees, bootstrap sampling, and max-features set to 'sqrt'.
- LR: logistic regression with L2 regularization.

State-of-the-art Baselines: Our baseline comparison includes the underlying black-box classification model's own (self-) *confidence scores*. Specifically, for all deep neural models, i.e., ResNET50, MLP, and CNN, we rely on the confidence score given by max class probability (DNN-MCP), as proposed by [16], which is a well established strong baseline. For RF's, we rely on the *uncertainty* score, computed as per the state-of-the-art method for random forest (RF-uncertainty), as proposed by [36]. For SVM, we rely on the standard approach of computing confidence scores over prediction probabilities from decision values after Platt scaling (SVM-Platt) [30]. For LR, the confidence score is given by the distance from the decision boundary (LR-Confidence).

Our main comparison is with the state-of-the-art method *Trust Score* [20]. Similar to *Risk Advisor*, Trust Score is a model-agnostic post-hoc approach, which takes as input a black-box classifier's predictions, and training data to produce point-wise trust scores for newly seen test points.

While calibrating a classifier's scores is a popular technique for producing calibrated confidence values, such techniques are rank-preserving. As all our evaluation metrics, i.e., AUROC, AUPR, and PRR (introduced later in Subsections V-B, V-D and V-C) are based on seeing different relative rankings of the scores rather than absolute values, there is no point in comparing against rank-preserving calibration techniques.

Implementation: The *Risk Advisor* is implemented as an ensemble of 10 SGBT classifiers, each initialized with a different random seed. Train and test sets are constructed using a 70:30 stratified split. All categorical features are one-hot encoded. Best hyper-parameters are chosen via grid-search by performing 5-fold cross validation. For *Risk Advisor's* E-SGBT model, we tune max-depth in [3,4,5,6], sample-rate in [0.25, 0.5, 0.75] and num-estimators in [100, 1000]. For *Trust Score*, we use the code shared by [20] and perform grid search over the parameter space reported in the paper. All experiments are conducted using scikit-learn and Keras on 2 GPUs and CPUs. Results reported are mean values over 5 runs.

B. Predicting Test-time Failure Risks

First, we evaluate to what extent the *Risk Advisor* can successfully detect test points misclassified by the underlying ML system. We measure the quality of failure prediction using standard metrics used in the literature [16]: area under ROC

Dataset	CIFAR 10	Fashion MNIST	MNIST	Census Income				Law School				Wine Quality white → white, red				Heart Disease US → US, UK, CH, HU				Census Income Male → Male, Female				Law School White → White, Black			
Black-box (BBox) classification model	ResNet50	CNN	CNN	LR	MLP	RF	SVM	LR	MLP	RF	SVM	LR	MLP	RF	SVM	LR	MLP	RF	SVM	LR	MLP	RF	SVM	LR	MLP	RF	SVM
LR-Confidence	-	-	-	**0.80**	-	-	-	0.70	-	-	-	0.62	-	-	-	0.70	-	-	-	0.78	-	-	-	0.71	-	-	-
SVM-Platt [30]	-	-	-	-	-	-	0.83	-	-	-	0.75	-	-	-	0.72	-	-	-	0.76	-	-	-	0.85	-	-	-	0.75
DNN-MCP [16]	0.78	0.90	**0.98**	-	**0.80**	-	-	-	0.76	-	-	-	0.57	-	-	-	0.68	-	-	-	0.78	-	-	-	0.76	-	-
RF-uncertainty [36]	-	-	-	-	-	0.75	-	-	-	0.71	-	-	-	0.65	-	-	-	**0.75**	-	-	-	0.69	-	-	-	0.70	-
Trust score [20]	0.64	0.88	0.96	0.65	0.65	0.71	0.71	0.69	0.69	0.83	0.81	**0.67**	0.71	0.69	0.73	0.70	0.66	0.65	0.69	0.62	0.62	0.67	0.67	0.70	0.68	0.83	0.82
Riskscore (Proposed)	**0.80**	**0.92**	**0.98**	**0.80**	**0.80**	**0.87**	**0.86**	**0.83**	**0.77**	**0.86**	**0.86**	0.66	**0.75**	**0.74**	**0.75**	**0.78**	**0.72**	0.72	**0.79**	**0.79**	**0.79**	**0.87**	**0.87**	**0.83**	**0.77**	**0.86**	**0.85**

TABLE I: AUROC for predicting test-time failure risks: Values in the table are area under ROC curve (AUROC). Higher values are better.

Dataset	CIFAR 10	Fashion MNIST	MNIST	Census Income				Law School				Wine Quality white → white, red				Heart Disease US → US, UK, CH, HU				Census Income Male → Male, Female				Law School White → White, Black			
Black-box (BBox) classification model	ResNet50	CNN	CNN	LR	MLP	RF	SVM	LR	MLP	RF	SVM	LR	MLP	RF	SVM	LR	MLP	RF	SVM	LR	MLP	RF	SVM	LR	MLP	RF	SVM
LR-Confidence	-	-	-	0.59	-	-	-	0.39	-	-	-	0.23	-	-	-	0.41	-	-	-	0.56	-	-	-	0.41	-	-	-
SVM-Platt [30]	-	-	-	-	-	-	0.66	-	-	-	0.51	-	-	-	0.44	-	-	-	0.52	-	-	-	0.69	-	-	-	0.50
DNN-MCP [16]	0.57	0.80	**0.96**	-	**0.60**	-	-	-	0.52	-	-	-	0.14	-	-	-	0.36	-	-	-	0.56	-	-	-	0.53	-	-
RF-uncertainty [36]	-	-	-	-	-	0.51	-	-	-	0.41	-	-	-	0.30	-	-	-	**0.50**	-	-	-	0.38	-	-	-	0.41	-
Trust score [20]	0.29	0.77	0.91	0.31	0.30	0.41	0.42	0.38	0.38	0.67	0.63	**0.34**	0.41	0.38	0.46	0.39	0.32	0.30	0.38	0.24	0.25	0.34	0.35	0.41	0.36	0.66	0.63
Riskscore (Proposed)	**0.59**	**0.83**	**0.96**	**0.61**	**0.60**	**0.74**	**0.73**	**0.65**	**0.54**	**0.73**	**0.71**	0.32	**0.50**	**0.47**	**0.50**	**0.57**	**0.45**	0.44	**0.57**	**0.57**	**0.58**	**0.74**	**0.74**	**0.66**	**0.53**	**0.73**	**0.70**

TABLE II: Risk mitigation by selective abstention. Values in the table are prediction rejection ratio (PRR). Higher values are better.

curve (AUROC) and area under precision recall curve (AUPR), where misclassifications are chosen as the positive class.

Results: Table I shows a comparison between the black-box models' own *confidence scores* [16, 30, 36], *Trust Score* [20] and the *Risk Advisor*'s estimated *risk score*, for all combinations of datasets and black-box models. Table I reports AUROC for detecting test-set errors of the underlying black-box classifiers. Best values are marked in bold. We make the following observations.

First, we observe that AUROC values for all the methods are higher than a random baseline (AUROC of 0.5), indicating that all the approaches are informative in detecting test errors. Second, the proposed *risk score* consistently *outperforms* black-box models' own confidence scores (barring a few exceptions). This holds true for all families of black-box classifiers including deep neural models and Random Forests, which build on DNN-MCP [16] and RF-uncertainty [36]. Finally, we observe that our Risk Advisor's *risk scores* consistently *outperform Trust Scores* by a significant margin, for all the datasets and all families of black-box classifiers. Similar trends hold for the AUPR metric (not shown for lack of space).

C. Application: Risk Mitigation by Selective Abstention

A benefit of predicting failure risks at deployment time is that we can take meaningful *risk mitigation* actions. For instance, if we expect that a ML system is likely to misclassify certain deployment/test-points, we can ask the ML system to abstain from making predictions and instead forward these data points to a fall-back system or human expert. In this experiment, we simulate the latter scenario as follows.

Setup and Metric: We generate a ranking of all the test points by ordering them according to the scores assigned by each approach, i.e., black-box model's *confidence score* (ascending order), *trust score* (ascending order), and *Risk Advisor*'s *risk score* (descending order), respectively. We then use these rankings to choose test points to defer to an *oracle*, in which case the ML systems predictions are replaced with the *oracle's* labels. This setup allows us to compute an *Accuracy-Rejection curve* (AR curve) [2, 9, 25]. AR curves are summarized using *prediction rejection ratio* (PRR), a metric which measures the degree to which the uncertainty scores are informative [25]. The *PRR* score lies between 0.0 and 1.0, where 1.0 indicates perfect ordering, and 0.0 indicates 'random' ordering.

Results: Table II shows a comparison between the black-box models' own *confidence scores* [16, 30, 36], *Trust Scores* [20] and the proposed *risk scores*. Values in the table are PRR values for all combinations of datasets and models. Best results are highlighted in bold. We make the following observations.

First, all methods under comparison have a PRR > 0, indicating that all the approaches are informative and better than a random baseline (with random abstention). Second, *risk scores* consistently yield the *best PRR* across all datasets and classification models (barring a few exceptions). There is no clear winner between Trust Scores and each of the black-box classifiers' native confidence scores.

D. Application: Detecting Out-of-distribution Test Points

In this experiment, we evaluate how well the *Risk Advisor* can successfully detect the underlying sources of uncertainty. However, for real-world datasets and complex black-box models it is difficult to collect ground truth (for evaluation) on which errors are due to inherent data complexity or model limitations. Hence, in this section we only focus on detecting errors due to *lack of knowledge* (e.g., due to data shifts between training and deployment distributions). To this end, we narrow our focus on the four datasets on out-of-distribution (OOD) test points for which we have ground truth labels shown in Table III. Our goal is to evaluate how well the *Risk Advisor's* estimated *epistemic uncertainty* can be used to detect test points coming from a different distribution than the one which the model was trained on.

Setup and Metric: Given a combined test dataset consisting of both in-distribution and out-of-distribution test points, the question at hand is to what extent the *Risk Advisor*'s estimated *epistemic uncertainty* can effectively separate in-distribution and out-of-distribution test points. As we have ground truth

Dataset	Wine Quality white → white, red				Heart Disease US → US, UK, CH, HU				Census Income Male → Male, Female				Law School White → White, Black			
BBox classification model	LR	MLP	RF	SVM	LR	MLP	RF	SVM	LR	MLP	RF	SVM	LR	MLP	RF	SVM
LR-Confidence	0.33	-	-	-	0.66	-	-	-	0.45	-	-	-	0.68	-	-	-
SVM-Platt [30]	-	-	-	0.81	-	-	-	0.67	-	-	-	0.42	-	-	-	0.66
MCP [16]	-	0.25	-	-	-	0.63	-	-	-	0.42	-	-	-	0.61	-	-
RF-epistemic uncertainty [36]	-	-	**0.84**	-	-	-	0.55	-	-	-	**0.64**	-	-	-	0.62	-
Trust score [20]	0.61	0.65	0.62	0.64	0.66	**0.65**	0.62	0.64	0.42	0.42	0.42	0.42	0.66	0.68	0.67	0.62
Epistemic uncertainty	**0.81**	**0.87**	0.82	**0.91**	**0.67**	0.54	**0.74**	**0.72**	**0.51**	**0.57**	0.54	**0.48**	**0.72**	**0.70**	**0.72**	**0.68**

TABLE III: Detecting out-of-distribution (OOD) test examples: Values in the table are AUROC for OOD detection. Higher values are better.

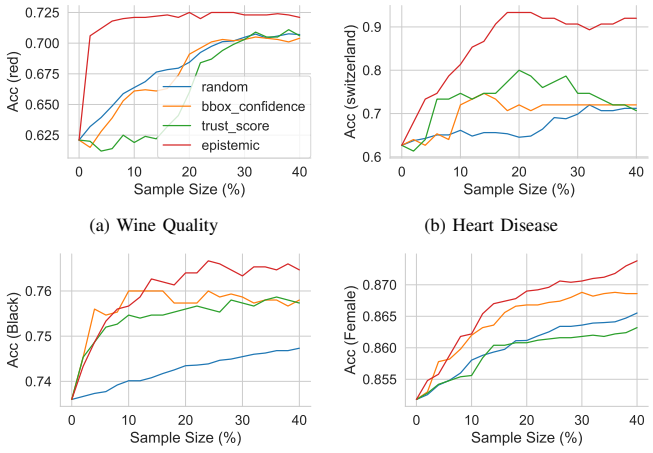
for out-of-distribution test points and we have ensured that there are equal numbers of in/out distribution test points, we can use the area-under-the-ROC-curve metric (AUROC) for evaluation. Intuitively, AUROC measures the degree to which each of the confidence scores ranks a randomly chosen OOD data point higher than a randomly chosen non-OOD point.

Results: Table III shows a comparison between the black-box model's own *confidence* score, *trust score*, and the *Risk Advisor*'s estimated *epistemic uncertainty*. Unlike baseline methods for DNN, SVM, and LR, the baseline for computing uncertainty of RF by [36] can decompose the overall uncertainty into aleatoric and epistemic components. Thus, for RF, we rely on the *epistemic* uncertainty estimates. We make the following observations.

First, observe that *epistemic uncertainty* consistently outperforms both the black-box model's own *confidence scores* and *trust scores* across all datasets and classification methods, with a significant margin. Further *Risk Advisor's* epistemic uncertainty is competitive with RF-epistemic uncertainty, which is model-specific. This supports our argument that a post-hoc *meta-learner* trained to compute uncertainties, is a viable alternative to replacing the underlying black-box ML classifier, which may not be feasible in production practice. Second, observe that for the Wine and Census Income datasets, the DNN-MCP[16] and LR *confidence score* has AUROC $<$0.5, i.e., a performance worse than the *random baseline*, implying that black-box model *incorrectly* assigns higher confidence scores for OOD points than for in-distribution points. A similar trend can be observed for *trust scores* for the Census Income dataset, thus indicating that *confidence* scores and *trust scores* are not that reliable under distribution shifts. In contrast, the the AUROC values for the *epistemic uncertainty* are always > 0.5, implying that the *Risk Advisor* always assigns higher *epistemic* uncertainty for OOD test points than for in-distribution test points. This is an important property, as it indicates that a ranked ordering of test points by *epistemic uncertainty* can be used in a deployed application to detect out-of-distribution test points (given an application-specific threshold). For these critical data points, the system could resort to a human expert (or other fall-back option), and thus enhance trustworthiness of the ML system.

E. *Application: Risk Mitigation by Sampling & Retraining*

Being able to identify black-box classifier's epistemic uncertainty enables another type of mitigation action: to mitigate risks due to evolving data by judiciously collecting more training examples and re-training the ML system.



(a) Wine Quality (b) Heart Disease

(c) Law School (White → Black) (d) Census Income (Male → Female)

Fig. 8: Addressing distribution shift: Comparison of various sampling strategies to selectively sample data points and retrain the black-box classification model. Curves that grow higher and faster from left to right are better.

We acknowledge the large body of literature on active sampling and domain adaptation in this context. In our experiment the goal is not to compare with these existing techniques, but rather to demonstrate an application of the *Risk Advisor*'s epistemic uncertainty, which can be achieved without making any changes to the underlying black-box classification system.

Setup: In this experiment, we fix the black-box classifier to logistic regression, and we assume that we have access to an untouched held-out set of labeled samples (different from training and test set). Our goal is to evaluate if the performance of the underlying black-box classifier can be improved for out-of-distribution test points by additional sampling and re-training the ML system on (a subset of) these held-out points. To evaluate the performance, we use the black-box classifier's improvement in accuracy for out-of-distribution test points.

We compare different sampling strategies by selecting data points from the with-held set in different orders based on three criteria: the *LR-Confidence*, *Trust score*, and the *Risk Advisor*'s *epistemic uncertainty*. For each approach, we first compute point-wise scores for all the points in the held-out set (different from training and test set, kept aside for sampling experiment). We then order the points in the held-out set according to these scores, i.e., *LR-confidence* (ascending order), *Trust score* (ascending order), and *Risk Advisor*'s *epistemic uncertainty* (descending order), respectively. Next, at each round of an iterative sampling, we select $k\%$ points from the held-out set (with replacement), and re-train the ML system.

Results: Fig. 8 shows results averaged over 5 independent

runs. The x-axis shows the percentage of additional points sampled from the held-out set for re-training, and the y-axis shows the corresponding improved accuracy for the OOD group (e.g., accuracy on red wine for the Wine dataset). Ideally, we would expect the accuracy to rise higher with as few additional training points as possible. We make the following observations.

First, as we sample and retrain on additional points from the held-out data, the accuracy for OOD test-points increases for all the approaches on all datasets. However, the percentage of additional samples required to achieve similar performance differs across approaches. Not surprisingly, *random* sampling is the slowest improving approach for 3 out of 4 datasets, followed by *trust scores* and *confidence scores*. The Risk Advisor's sampling by *epistemic* uncertainty consistently outperforms on all datasets, by a large margin. For instance, on the Heart Disease dataset *epistemic* uncertainty achieves 30 percentage points (pp) improvement in accuracy (from 0.6 to 0.9) for an additional 20% samples from the held-out set. In contrast, all the other approaches stagnate around 0.7 even for an additional 40% samples. Similarly, on the Wine Quality dataset we see an improvement of 10 pp for an additional 10% samples, while other approaches do not reach this improvement even for additional 40% of samples. We observe similar trends across approaches for Law School and Census Income datasets, albeit with smaller gains.

VI. Conclusion

This paper presented the *Risk Advisor* model for detecting and analyzing sources of uncertainty and failure risks when a trained classifier is deployed for production usage. In contrast to prior works, the Risk Advisor treats the base classifier as a black-box model, and this model-agnostic approach makes it a highly versatile and easy-to-deploy tool. In contrast to the prior state-of-the-art (including the main baseline Trust Scores [20]), the Risk Advisor goes beyond providing a single measure of uncertainty, by computing refined scores that indicate failure risks due to data variability and noise, systematic data shifts between training and deployment, and model limitations. Extensive experiments on real-world and synthetic datasets covering common ML failure scenarios show that the Risk Advisor reliably predicts deployment-time failure risks in all the scenarios, and outperforms strong baselines. Thereby, we believe the Risk advisor, with its ability to proactively audit and identify potential regions of failure risks would be a useful asset for the trustworthy machine learning toolbox.

VII. Acknowledgment

This research was supported by the ERC Synergy Grant "imPACT" (No. 610150) and ERC Advanced Grant "Foundations for Fair Social Computing" (No. 789373).

References

[1] D. Barber and C. M Bishop. 1998. Ensemble learning in Bayesian neural networks. *J. Comput. Syst. Sci.* 168 (1998).
[2] P. L Bartlett and M. H Wegkamp. 2008. Classification with a reject option using a hinge loss. *JMLR* 9, Aug (2008).
[3] Irad Ben-Gal. 2005. Outlier detection. In *Data mining and knowledge discovery handbook*. Springer.
[4] Thomas M Cover. 1999. *Elements of information theory*. John Wiley & Sons.
[5] J. S Denker and Y. LeCun. 1990. Transforming neural-net output levels to probability distributions. In *NeurIPS*.
[6] S. Depeweg et al. 2018. Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning. In *ICML*.
[7] A. Der Kiureghian and O. Ditlevsen. 2009. Aleatory or epistemic? Does it matter? *Structural safety* 31, 2 (2009).
[8] D. Dua and C. Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml
[9] R. El-Yaniv et al. 2010. On the Foundations of Noise-free Selective Classification. *JMLR* 11, 5 (2010).
[10] R. M Fano. 1961. Transmission of information: A statistical theory of communications. *American Journal of Physics* 29, 11 (1961).
[11] J. H Friedman. 2002. Stochastic gradient boosting. *Computational statistics & data analysis* 38, 4 (2002).
[12] Y. Gal and Z. Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*.
[13] I. J Goodfellow, J. Shlens, and C. Szegedy. 2015. Explaining and harnessing adversarial examples. In *ICLR*.
[14] C. Guo et al. 2017. On Calibration of Modern Neural Networks. In *ICML*.
[15] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep residual learning for image recognition. In *CVPR*.
[16] D. Hendrycks and K. Gimpel. 2017. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *ICLR*.
[17] A. Honkela and H. Valpola. 2004. Variational learning and bits-back coding: an information-theoretic view to Bayesian learning. *IEEE Trans. Neural Netw.* 15, 4 (2004).
[18] Stephen C Hora. 1996. Aleatory and epistemic uncertainty in probability elicitation with an example from hazardous waste management. *Reliability Engineering & System Safety* 54, 2-3 (1996).
[19] E. Hüllermeier and W. Waegeman. 2021. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning* 110, 3 (2021).
[20] H. Jiang, B. Kim, M. Y Guan, and M. R Gupta. 2018. To Trust Or Not To Trust A Classifier.. In *NeurIPS*.
[21] A. Kendall and Y. Gal. 2017. What uncertainties do we need in Bayesian deep learning for computer vision?. In *NeurIPS*.
[22] A. Krizhevsky. 2009. *Learning multiple layers of features from tiny images*. Technical Report.
[23] B. Lakshminarayanan, A. Pritzel, and C. Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*.
[24] Yann LeCun, Corinna Cortes, and CJ Burges. 2010. MNIST handwritten digit database. *Available: http://yann.lecun.com/exdb/mnist* 2 (2010).
[25] A. Malinin. 2019. Uncertainty Estimation in Deep Learning with application to Spoken LanguageAssessment. *PhD thesis* (2019).
[26] A. Malinin, L. Prokhorenkova, and A. Ustimenko. 2020. Uncertainty in gradient boosting via ensembles. *arXiv preprint arXiv:2006.10562*.
[27] A. Nguyen, J. Yosinski, and J. Clune. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*.
[28] Y. Ovadia et al. 2019. Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. *NeurIPS*.
[29] F. Pedregosa et al. 2011. Scikit-learn: ML in Python. *JMLR* 12 (2011).
[30] J. Platt et al. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers* 10, 3 (1999).
[31] S Saria and A. Subbaswamy. 2019. Tutorial: safe and reliable machine learning. *arXiv preprint arXiv:1904.07204* (2019).
[32] S. Schelter, T. Rukat, and F. Biessmann. 2020. Learning to Validate the Predictions of Black Box Classifiers on Unseen Data. In *SIGMOD*.
[33] S. Schneider, E. Rusak, L. Eck, and O. et al. Bringmann. 2020. Improving robustness against common corruptions by covariate shift adaptation. *In NeurIPS*.
[34] P. Schulam and S. Saria. 2019. Can you trust this prediction? Auditing pointwise reliability after learning. In *AISTATS*.
[35] R. Senge, S. Bösner, K. Dembczyński, J. Haasenritter, and O. et al. Hirsch. 2014. Reliable classification: Learning classifiers that distinguish aleatoric and epistemic uncertainty. *Information Sciences* 255 (2014).
[36] M. H. Shaker and E. Hüllermeier. 2020. Aleatoric and epistemic uncertainty with random forests. In *IDA*.

[37] I. Steinwart, D. Hush, and C. Scovel. 2005. A Classification Framework for Anomaly Detection. *JMLR* 6, 2 (2005).

[38] L. F Wightman. 1998. LSAC National Longitudinal Bar Passage Study. LSAC Research Report Series. (1998).

[39] H. Xiao, K. Rasul, and R. Vollgraf. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* (2017).