Towards Algorithm-Agnostic Uncertainty Estimation: Predicting Classification Error in an Automated Machine Learning Setting

Matthias König H.M.T.KONIG@LIACS.LEIDENUNIV.NL
Holger H. Hoos HH@LIACS.NL
Jan N. van Rijn J.N.VAN.RIJN@LIACS.LEIDENUNIV.NL
Leiden Institute of Advanced Computer Science, Leiden University, The Netherlands

Abstract

Detecting and signaling when a machine learning algorithm, such as a classifier, makes erroneous predictions is of crucial importance for the development of trustworthy AI systems. At the same time, most work on uncertainty estimation tends to be limited to a specific type of predictor or only considers regression tasks. In this work, we present a flexible method for estimating uncertainty in classification procedures using several, classifier-independent measures that act as proxies for the uncertainty associated with predictions. Our approach yields promising results on a variety of benchmark datasets and can be used alone or in combination with the uncertainty estimates produced by the classifier.

Keywords: Classification, Uncertainty Estimation, Predictive Performance Modeling

1. Introduction

As AI-based systems are becoming an increasingly important tool for individuals and practitioners within various domains and use contexts (Barr and Feigenbaum, 2014), there need to be mechanisms in place that assess the reliability of such systems. This is important, as even the best machine learning methods are expected to produce imperfect results. Common approaches to evaluating machine learning algorithms include holdout and cross-validation strategies, where the learned predictor is applied to and validated on a previously unseen partition of the data set at hand. However, once a system has been deployed, such testing mechanisms no longer guarantee reliability. For example, there might be cases, where incoming data deviates from the distribution of the data the system has been trained on (Gama et al., 2004). At the same time, it may be difficult to verify the correctness of predictions when there is no access to ground truth. If a classifier produces incorrect predictions and there is no way to identify such errors, this could have dangerous consequences. For example, a classifier used in medical diagnostics producing an erroneous output might cause serious physical harm; furthermore, it could foster distrust in AI-based technologies in general. Detecting and signaling when a classifier is in error is therefore a crucial component of the development of trustworthy AI systems.

There are various causes of classification error. For example, individual outliers are often misclassified (Abe et al., 2006). Border points, that is, points close to an inter-class boundary, also tend to cause classification error (Brighton and Mellish, 2002). Further potential sources of error could be named; these and others motivate the goal of this work: Developing measures that **(*i*) can be used to identify data points that are likely to be**

©2020 Matthias König, Holger H. Hoos and Jan N. van Rijn.

misclassified (i.e., that the predictor should be uncertain about) and ***(ii)* are independent of the classification algorithm being used**, which we also refer to as our *base classifier*. The latter point is motivated by two observations. Firstly, most previous work on uncertainty estimation is rather inflexible as it tends to be limited to a specific type of base classifier, for example deep neural networks (Mandelbaum and Weinshall, 2017; Kendall and Gal, 2017; Hendrycks and Gimpel, 2016). In contrast, when training an AutoML system, many machine learning algorithms are usually considered and selected according to their ability to solve the task at hand. Our method for estimating uncertainty therefore does not link to any specific algorithm, but seeks to be generically applicable. Secondly, previous work on uncertainty estimation that goes beyond deep models focuses on regression rather than classification tasks (Liu et al., 2019; Duan et al., 2019). One explanation for this is that "probabilistic estimation is already the norm in classification problems" (Duan et al., 2019). However, this statement is somewhat problematic. For example, a random forest classifier determines probability as the fraction of trees in a forest that vote for a certain class, although this has been shown to not always produce reasonable estimates for uncertainty (Olson and Wyner, 2018). To the best of our knowledge, there is no work that analyses the ability of such probability scores to capture uncertainty.

In brief, the main contributions of our work presented in the following are: (i) Several measures that act as proxies for classifier uncertainty and capture data point characteristics such as class likelihood or overlap, which may cause classification error; (ii) promising results when using these measures for the prediction of classification error across a wide range of datasets, indicating a significant relative improvement over the baseline by 55%.

2. Related Work

When we are to model uncertainty of a classifier, we implicitly model its performance. That is, we are interested in the question of whether we can we predict if a trained classifier will produce an error on a particular held-out test example. This stands in contrast to methods that aim at identifying erroneous data instances in the training set (John, 1995; Brodley and Friedl, 1999). While such approaches can improve the quality of the training data at hand, they do not provide mechanisms to deal with misclassification of previously unseen data points, once a trained classifier has been deployed.

The question of whether we can predict classification error on an unseen data instance can be placed within the context of empirical performance modeling. An empirical performance model is a model that predicts the performance of algorithms on previously unseen input, including previously unseen problem instances. Such models can and have been used to, for example, predict the running time distribution of an unseen algorithm configuration on a given problem instance (Hutter et al., 2014; Eggensperger et al., 2018). The main idea behind these models is to empirically learn a function that maps properties of the input, i.e., the problem instance or data point, to the desired performance metric. Against this background, it is not surprising that empirical performance models have also been used to predict the expected accuracy of a given algorithm on an unseen dataset, as done in work by Garcia et al. (2018). Here, the task is to predict the accuracy of different classifiers and recommend the best classifier to use for a particular classification problem, i.e., dataset. Their work builds upon the idea of meta-learning (Brazdil et al., 2008) and maps global dataset features to the predictive performance of the classifiers. We deploy a

somewhat similar framework, in which we map characteristics of a single data instance to the predictive performance of a given classifier.
The idea of mapping data point attributes to the predictive accuracy of a classifier is not entirely new and has previously been introduced in the form of grading classifiers (Seewald and Fürnkranz, 2001). Here, data instances are labeled with predictions that have been marked as correct or incorrect. Subsequently, these labels are used as targets for a meta-classifier that predicts when a given base classifier will produce incorrect results. While we also 'grade' instances at the base level, we do not use the data point attributes as our training corpus, but develop features, i.e., *uncertainty measures*, specifically tailored for capturing potential sources of classification errors.

3. Methodology

The goal of this work is to learn a meta-classifier that can predict, based on a set of uncertainty measures whether an instance will be subject to classification error or not. This meta-classifier will generally be referred to as the *uncertainty model*.

3.1 General Framework

Training the Base Classifier. We start by randomly splitting each dataset into training, validation and testing partitions; the partition sizes are 56%, 19% and 25% of the full dataset, respectively. The test set remains untouched until we test our system for uncertainty prediction. After splitting the data, we train a base classifier using an AutoML approach; more specifically, we employ the auto-sklearn package (Feurer et al., 2015). All hyperparameters are set to their default values, but we limit the search space by only including random forest and gradient boosting classifiers. Thereby, we reduce computational expenses, while still achieving good performance on almost every dataset. The time limit is set to 7200 seconds per task. Next, we use the base classifier to generate predictions for examples in the set. Subsequently, we compare these predictions to the ground truth labels and produce a new, binary label indicating whether an instance has been correctly classified or not. This binary label forms the new target class labels for the following training step, that is, the training of the uncertainty model. At the same time, we compute uncertainty measures and use those as training data for the uncertainty model. An overview of this approach can be found in Appendix B.1.
Training the Uncertainty Model. There are two modes for training the uncertainty model. In the first mode, we train an uncertainty model for every dataset individually. Thus, a meta-classifier is fitted on the uncertainty measures computed for each respective dataset. In the second training mode, we follow a *leave-one-dataset-out* protocol, where the computed uncertainty measures and binary labels of all n-1 datasets or more specifically, validation sets, serve as training data. On the other hand, the uncertainty measures computed for the remaining dataset serve as testing data for the uncertainty model. The idea behind this approach is to learn a representation for uncertainty by using information from all available datasets, rather than considering each dataset in isolation. For training of the uncertainty model, we again employ auto-sklearn and follow the same training protocol as used for the base classifier, except for the objective metric, which we change to the F1 score with incorrectly classified examples set as the positive class. The F1 score can

be interpreted as a weighted harmonic mean of precision and recall and thus prevents the uncertainty model from being optimised towards only one of these metrics, as that might be at the expense of the other one.

3.2 Uncertainty Measures

To describe an instance, we need to compute features that provide information about the likelihood of that instance to be misclassified. A straightforward approach would be to make a probabilistic estimation, for example, by obtaining the class probability provided by the given classifier. We refer to this probability as the *classifier score* and use it as our baseline when predicting classification error. As previously reported, the classifier score alone is not expected to provide reliable information when it comes to model uncertainty. Therefore, we include further uncertainty measures, which are defined in the following paragraphs. It should be noted that these measures can be computed with little computational overhead; more specifically, we computed them within less than 30 seconds for every dataset.

k-Disagreeing Neighbours. The k-disagreeing neighbours (kDN) uncertainty measure builds on work by Smith et al. (2014). In their work, they present complexity measures that capture the 'hardness' of a data instance, which can be interpreted as a proxy for the likelihood of a single data point to be misclassified. One of these measures is the kDN measure that is computed as the fraction of the k nearest neighbours for an instance that do not share its target class value. Since we assume the absence of ground truth labels when classifying unseen data points, we make a small modification and replace the target class value by the predicted class value for instance x:

$$k\text{DN}(x) = \frac{|\{y : y \in k\text{NN}(x) \wedge y \neq \hat{y}\}|}{k} \tag{1}$$

where kNN(x) is the set of k nearest neighbours of instance x and $\hat{y}$ is the predicted class.

k-Disagreeing Classifiers. The k-disagreeing classifiers (kDC) uncertainty measure is inspired by the notion of landmarking (Pfahringer et al., 2000). That is, we characterise an observation by the predictions of a set of auxiliary classifiers or *landmarkers*. As we would like those classifiers to measure different properties of a data instance, we choose them from a broad range of algorithms; more specifically, we use every classifier available in auto-sklearn with default settings (Feurer et al., 2015). In order to ensure the landmarker's ability to learn a somewhat meaningful representation of the input data, we only consider predictions by classifiers that achieve an accuracy of at least 60% on the test set. Formally, the kDC measure can be defined as:

$$kDC(x) = \frac{|\{y : y \in G(x) \wedge y \neq \hat{y}\}|}{k} \tag{2}$$

where *G(x)* is a set of predictions made by auxiliary classifiers on the instance x and $\hat{y}$ is the predicted class for instance x made by the base classifier.

Class Likelihood. Class likelihood (CL) measures the likelihood of an instance belonging to the predicted class. We define the Class Likelihood measure as:

$$P(\hat{y}) \mid x_1, ..., x_n) \propto P(\hat{y}) \cdot \prod_{i=1}^{n} P(x_i \mid \hat{y}) \tag{3}$$

where $\hat{y}$ refers to the predicted class and and x_i is the value of instance x's i-th attribute. In practice, we implement the CL measure by means of a naïve bayes classifier.

Distance to the Closest Centroid. The distance to the closest centroid (DCC) represents a measure to detect whether an instance might be off the training distribution. The DCC measure is described by the Euclidean distance of the observed instance to the centre point of the closest k-means cluster. The number of clusters is computed according to the number of classes present in the data.

3.3 Data

Experiments have been performed on the OpenML-CC18 benchmark suite (Bischl et al., 2017), which comprises 72 handpicked classification datasets from a broad range of domains, ranging from plant classification to loan approval. Since convolutional neural networks are the state-of-the-art for image classification datasets, we excluded these datasets, which leaves us with 57 datasets. A list of all datasets and their corresponding characteristics, such as the number of features and samples, can be found in Appendix A.1. We further remove datasets where a) the base classifier achieves close to perfect accuracy on the test set, i.e., where the number of misclassified instances is smaller than 5, thereby not allowing for efficient training of the uncertainty model and/or b) the base classifier predicted the same class for each sample in the test set, i.e., where it did not learn a meaningful representation of the data. Thereby, 12 datasets were eliminated, resulting in a final set of 45 datasets.

4. Results

For all experiments, we report performance metrics averaged across all used datasets in the form of F1 score, precision and recall along with their standard deviation. As these aggregate statistics may not reflect individual dataset performance, we also provide box plots of the distribution of the F1 metric. These can be found in Appendices B.2 and B.3. Differences in performance metrics are tested for statistical significance using permutation test with the number of permutations set at 10 000 and a significance threshold of 0.05.

Results on individual datasets. Firstly, we test for the performance of each uncertainty measure alone and compare their individual informative value. While all measures, except for the class likelihood, show slightly better performance in terms of the F1 score, these differences are not statistically significant. However, when all measures are merged and used as joint input for the uncertainty model we achieve the highest recall and significantly outperform the baseline (Table 1). This observation suggests that we are able to detect significantly more misclassified instances than if we were to only use the classifier score.

Results from leave-one-dataset-out cross-validation. In order to ensure a fair comparison, we first compute a new baseline using only the classifier score. Subsequently, we add every uncertainty measure to the model and achieve significantly better performance in terms of both the F1 score and recall (Table 2), while precision remains statistically unchanged. This indicates that, when learned from a large number of observations, our measures are able to capture uncertainty beyond the classifier's probabilistic score. It should be noted that this improvement can be found consistently across datasets, rather than just for a few outliers. This is visualised in Appendix B.4. Lastly, we fit an uncertainty

Table 1: **Averaged results on individual datasets, where for each dataset, an individual uncertainty model is trained.** In columns marked with **[C]**orrect, performance metrics are reported with correctly classified instances treated as the positive class. Boldfaced values represent those that are significantly higher than the baseline.

Uncertainty Measure	F1	Precision	Recall	Precision [C]	Recall [C]
Classifier Score (Baseline)	0.40 ± 0.18	0.51 ± 0.24	0.36 ± 0.18	0.84 ± 0.19	0.91 ± 0.12
Class Likelihood	0.34 ± 0.17	0.48 ± 0.25	0.31 ± 0.18	0.74 ± 0.21	0.87 ± 0.17
Distance to Closest Centroid	0.43 ± 0.13	0.54 ± 0.19	0.38 ± 0.13	0.79 ± 0.19	0.86 ± 0.17
k-Disagreeing Classifiers	0.44 ± 0.15	0.57 ± 0.19	0.38 ± 0.15	0.75 ± 0.23	0.87 ± 0.17
k-Disagreeing Neighbours	0.48 ± 0.14	0.58 ± 0.18	0.43 ± 0.14	0.79 ± 0.20	0.88 ± 0.17
Score + kDC + kDN + CL + DCC	0.48 ± 0.14	0.57 ± 0.19	$\mathbf{0.45} \pm 0.16$	0.79 ± 0.20	0.86 ± 0.20

Table 2: **Averaged results from leave-one-dataset-out cross-validation, where an uncertainty model is trained on n-1 datasets.** In columns marked with **[C]**orrect, performance metrics are reported with correctly classified instances treated as the positive class. Boldfaced values represent those that are significantly higher than the baseline.

Uncertainty Measure	F1	Precision	Recall	Precision [C]	Recall [C]
Classifier Score (Baseline)	0.38 ± 0.13	0.74 ± 0.20	0.28 ± 0.12	0.84 ± 0.26	0.91 ± 0.09
kDC + kDN + CL + DCC	$\mathbf{0.54} \pm 0.12$	0.75 ± 0.17	$\mathbf{0.43} \pm 0.12$	0.83 ± 0.15	0.96 ± 0.04
Score + kDC + kDN + CL + DCC	$\mathbf{0.59} \pm 0.14$	0.82 ± 0.18	$\mathbf{0.49} \pm 0.14$	0.84 ± 0.17	0.97 ± 0.03

model with the classifier score excluded. Though this approach achieves slightly lower performance, it still significantly outperforms the baseline. Most interestingly, it suggests that we can detect misclassified instances without accessing the properties of the base classifier, but rather in a universal fashion.

5. Conclusions and Future Work

We presented several measures that act as proxies for model uncertainty and capture data point characteristics, such as class likelihood or overlap, which in turn may cause classification errors. These measures can be computed irrespective of the base classifier and without access to ground truth, making them suitable for an online setting where access to ground truth is costly and/or subject to substantial delays. Beyond that, we find promising results when using these measures for classification error prediction across a wide range of datasets from the OpenML-CC18 benchmark suite. Most importantly, we show that each measure embodies information on model uncertainty and can be used in addition to uncertainty estimates produced by the classifier, which we argue does not suffice to reliably detect misclassified data instances. Furthermore, we show that the uncertainty measures transfer well across datasets, which supports their generality, and allows for more efficient training.
There are various avenues for further research. One possible path goes towards the inclusion of a broader range of techniques to be used as base classifiers. Furthermore, we assume that the set of uncertainty measures can be further extended, as there are probably more factors at play than the ones considered in this study. This is underlined by the observation that for some datasets, our uncertainty model achieves only limited performance. In light of this, we consider our results a promising first step towards making classification algorithms and their applications more reliable and trustworthy.

References

N. Abe, B. Zadrozny, and J. Langford. Outlier detection by active learning. In *Proc. 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 504–509, 2006.

A. Barr and E. A. Feigenbaum. *The Handbook of Artificial Intelligence: Volume 2.* Butterworth-Heinemann, 2014.

B. Bischl, G. Casalicchio, M. Feurer, F. Hutter, M. Lang, R. G. Mantovani, J. N. van Rijn, and J. Vanschoren. Openml benchmarking suites. *arXiv preprint arXiv:1708.03731*, 2017.

P. Brazdil, C. G. Carrier, C. Soares, and R. Vilalta. *Metalearning: Applications to data mining.* Springer Science & Business Media, 2008.

H. Brighton and C. Mellish. Advances in instance selection for instance-based learning algorithms. *Data Mining and Knowledge Discovery*, 6(2):153–172, 2002.

C. E. Brodley and M. A. Friedl. Identifying mislabeled training fata. *Journal of Artificial Intelligence Research*, 11:131–167, 1999.

T. Duan, A. Avati, D. Y. Ding, S. Basu, A. Y. Ng, and A. Schuler. Ngboost: Natural gradient boosting for probabilistic prediction. *arXiv preprint arXiv:1910.03225*, 2019.

K. Eggensperger, M. Lindauer, and F. Hutter. Neural networks for predicting algorithm runtime distributions. *Proc. 27th International Joint Conference on Artificial Intelligence*, 2018.

M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems*, pages 2962–2970, 2015.

J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with drift detection. In *Proc. Brazilian Symposium on Artificial Intelligence*, pages 286–295. Springer, 2004.

L. P. Garcia, A. C. Lorena, M. C. de Souto, and T. K. Ho. Classifier recommendation using data complexity measures. In *Proc. 24th International Conference on Pattern Recognition (ICPR)*, pages 874–879. IEEE, 2018.

D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

F. Hutter, L. Xu, H. H. Hoos, and K. Leyton-Brown. Algorithm runtime prediction: Methods & evaluation. *Artificial Intelligence*, 206:79–111, 2014.

G. H. John. Robust decision trees: Removing outliers from databases. In *KDD*, volume 95, pages 174–179, 1995.

A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems*, pages 5574–5584, 2017.

J. Liu, J. Paisley, M.-A. Kioumourtzoglou, and B. Coull. Accurate uncertainty estimation and decomposition in ensemble learning. In *Advances in Neural Information Processing Systems*, pages 8950–8961, 2019.

A. Mandelbaum and D. Weinshall. Distance-based confidence score for neural network classifiers. *arXiv preprint arXiv:1709.09844*, 2017.

M. A. Olson and A. J. Wyner. Making sense of random forest probabilities: a kernel perspective. *arXiv preprint arXiv:1812.05792*, 2018.

B. Pfahringer, H. Bensusan, and C. G. Giraud-Carrier. Meta-learning by landmarking various learning algorithms. In *Proc. 17th International Conference on Machine Learning*, pages 743–750, 2000.

A. K. Seewald and J. Fürnkranz. An evaluation of grading classifiers. In *International Symposium on Intelligent Data Analysis*, pages 115–124. Springer, 2001.

M. R. Smith, T. Martinez, and C. Giraud-Carrier. An instance level analysis of data complexity. *Machine Learning*, 95(2):225–256, 2014.

Appendix A. Tables

Table A.1: Datasets used from the OpenML-CC18 benchmark suite.

Dataset	**# instances**	**# features**
Bioresponse	3751	1777
Internet-Advertisements	3279	1559
cnae-9	1080	857
isolet	7797	618
har	10299	562
madelon	2600	501
dna	3186	181
nomao	34465	119
MiceProtein	1080	82
analcatdata_authorship	841	71
ozone-level-8hr	2534	73
splice	3190	61
spambase	4601	58
first-order-theorem-proving	6118	52
connect-4	67557	43
qsar-biodeg	1055	42
texture	5500	41
cylinder-bands	540	40
pc4	1458	38
pc3	1563	38

kr-vs-kp	3196	37
GesturePhaseSegmentationProcessed	9873	33
wdbc	569	31
PhishingWebsites	11055	31
sick	3772	30
steel-plates-fault	1941	28
wall-robot-navigation	5456	25
kc2	522	22
kc1	2109	22
pc1	1109	22
jm1	10885	22
numerai28.6	96320	22
climate-model-simulation-crashes	540	21
churn	5000	21
credit-g	1000	21
segment	2310	20
eucalyptus	736	20
vehicle	846	19
letter	20000	17
bank-marketing	45211	17
credit-approval	690	16
adult	48842	15
dresses-sales	500	13
vowel	990	13
ilpd	583	11
breast-w	699	10
tic-tac-toe	958	10

Appendix B. Figures

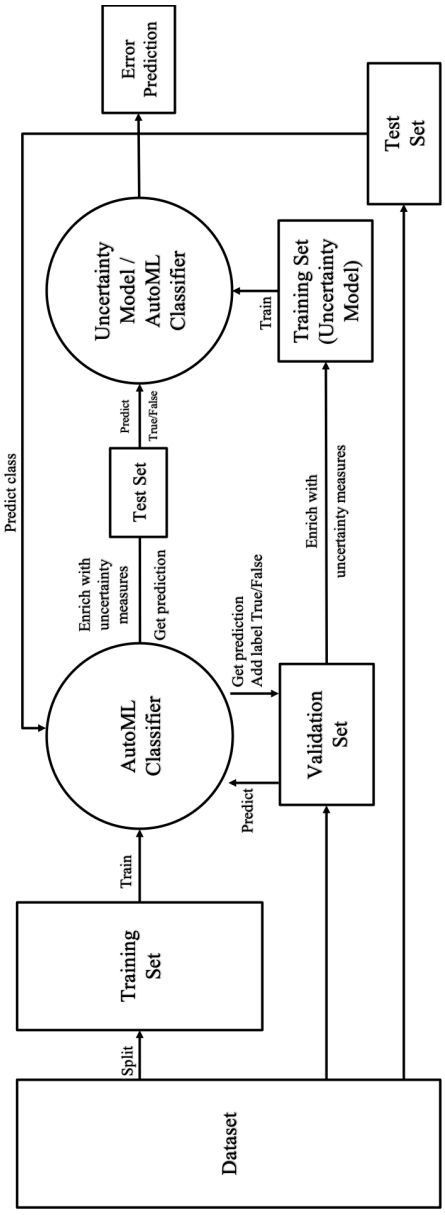

Figure B.1: **Schematic view of the technical framework:** For each sample from the test set, we compute binary labels and uncertainty measures that subsequently serve as input for the uncertainty model. For every holdout sample we compute uncertainty measures and use those to test the uncertainty model, which predicts whether an instance is likely to be misclassified.

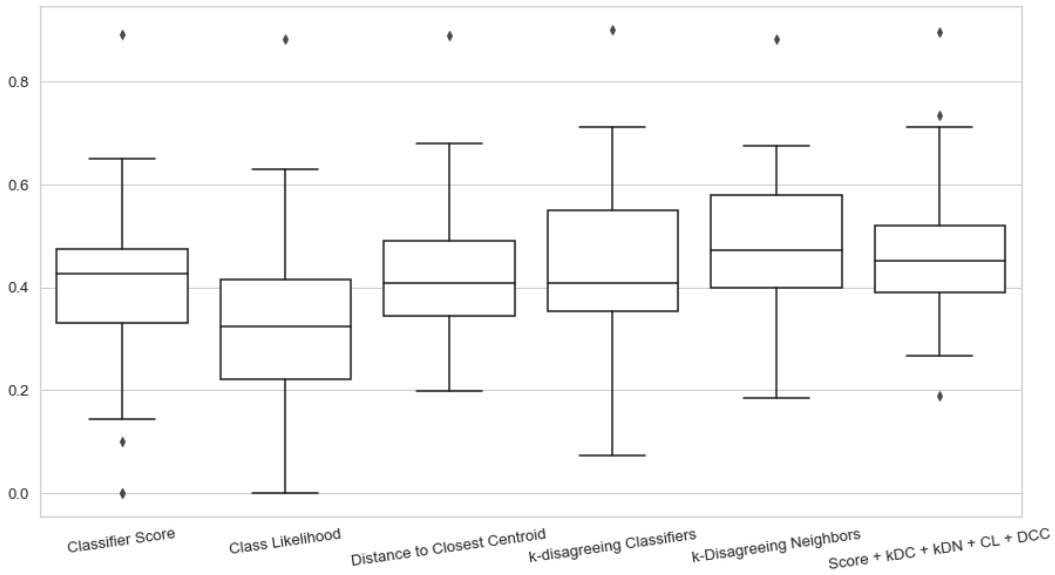

Figure B.2: Variance of the F1 metric for different measures when training an uncertainty model on each dataset individually.

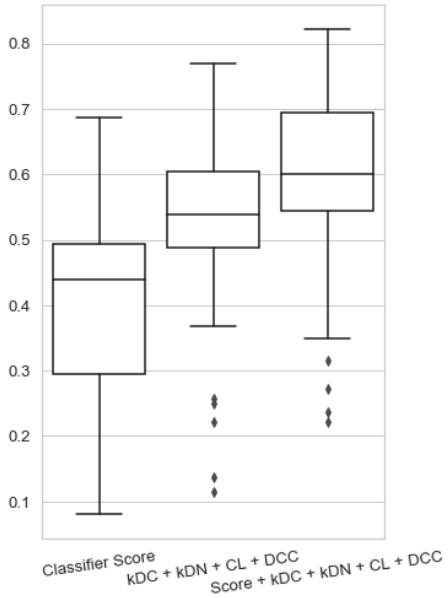

Figure B.3: Variance of the F1 metric for different measures when including n-1 datasets into uncertainty model training and testing on the remaining one; i.e., leave-one-dataset-out cross-validation.

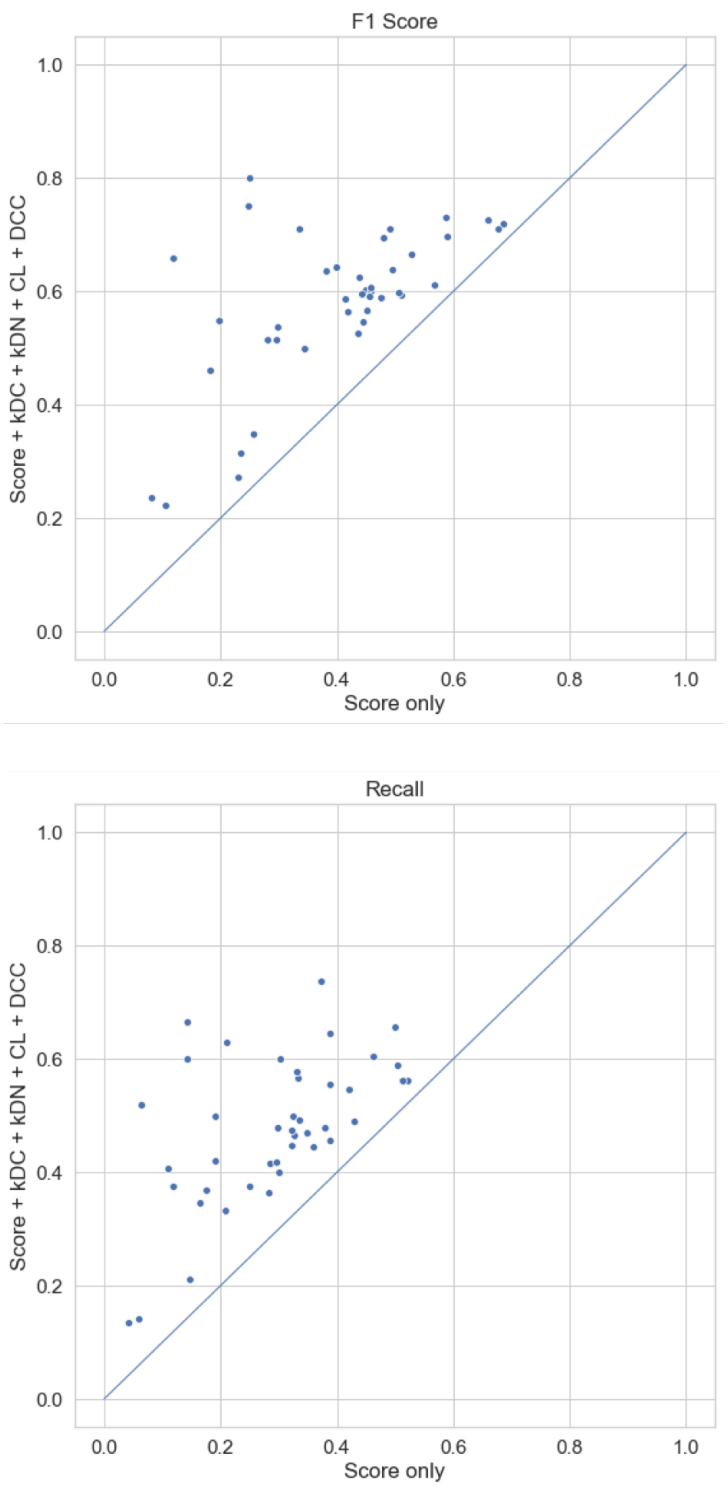

Figure B.4: Dataset-wise comparison of F1 score and recall values obtained from leave-out-dataset-out cross-validation, using (*i*) the classifier score only and (*ii*) the classifier score along with the uncertainty measures.