EXPLORING LOSS DESIGN TECHNIQUES FOR DECISION TREE ROBUSTNESS TO LABEL NOISE

Lukasz Sztukiewicz*, **Jack H. Good, Artur W. Dubrawski**
Auton Lab, Robotics Institute, Carnegie Mellon University
{lsztukie, jhgood, awd}@andrew.cmu.edu

ABSTRACT

In the real world, data is often noisy, affecting not only the quality of features but also the accuracy of labels. Current research on mitigating label errors stems primarily from advances in deep learning, and a gap exists in exploring interpretable models, particularly those rooted in decision trees. In this study, we investigate whether ideas from deep learning loss design can be applied to improve the robustness of decision trees. In particular, we show that loss correction and symmetric losses, both standard approaches, are not effective. We argue that other directions need to be explored to improve the robustness of decision trees to label noise.

INTRODUCTION AND RELATED WORK

Label errors, also known as label noise, are widespread. Recent estimates indicate that commonly used research data, which is usually assumed to be clean, contains up to 10% label corruption (Northcutt et al., 2021). This problem has received considerable attention in the deep learning community, resulting in several techniques to improve robustness of neural networks to incorrect labels. These methods are surveyed in Han et al. (2021) and Song et al. (2023). Despite these efforts, work on interpretable machine learning methods, such as decision trees, remains limited. Existing research highlights the resilience of decision trees to symmetric label noise in specific scenarios, particularly in binary classification with large sample sizes Ghosh et al. (2017). However, broader scenarios and enhancements in this context remain understudied. Most efforts involving tree-based algorithms focus on improving robustness of ensembles such as Random Forest (Yang et al., 2019; Zhou et al., 2019) or gradient boosting (Miao et al., 2016), with little attention given to individual decision trees. This raises a natural question: Can recent advances in deep learning be applied to conventional decision tree induction? Recognizing the limited exploration of this research direction Nanfack et al. (2022), our study aims to investigate this question. In particular, we investigate whether loss design approaches including loss correction and symmetric loss functions can be effectively adapted to tree-based algorithms.

THEORETICAL ANALYSIS

We consider c-class classification problems. Let $\mathcal{X} \in \mathbb{R}^m$ be the feature space and $\mathcal{Y} \in \mathbb{R}^c$ be the label space. Let $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ be random variables from which we observe the data set $\{(x_i, y_i)\}_{i=1}^n$, where each x_i is a vector representing m features and y_i is a one-hot encoded vector label indicating one of c classes. The observed data are assumed to be affected by noise; each (x_i, y_i) is drawn from a distribution $(X, \tilde{Y})$ which is related to the true labels by the noise transition matrix $T(x) \in [0, 1]^{c \times c}$, where $T_{a,b}(x) = P(\tilde{Y} = b | Y = a, X = x)$.

A loss function $\mathcal{L}$ maps a true and predicted label to a loss value; learning searches for a model that minimizes the empirical loss $\frac{1}{n}\sum_{i=1}^n \mathcal{L}(y_i, \hat{y}_i)$ for predictions $\hat{y}_i$. Decision tree learning using Gini gain or information gain minimizes the empirical mean squared error or cross-entropy loss, respectively, as shown in Appendix A.1. For both, the loss-minimizing values $v_\ell \in \mathbb{R}^c$ for leaves ℓ are the weighted means $v_\ell = \arg\min_v \sum_{i=1}^n \mu_\ell(x_i)\mathcal{L}(y_i, v) = \sum_{i=1}^n \mu_\ell(x_i) y_i / (\sum_i^n \mu_\ell(x_i))$ where μ_ℓ is the data membership function in leaf ℓ, taking values in $\{0, 1\}$.

Impurity-based tree growth is invariant to forward loss correction. Loss correction (Patrini et al., 2017) assumes the transition matrix T is known and incorporates it into the loss function so

*Preferred contact: research@lukaszsztukiewicz.com

that the loss minimizer is, in expectation, the same as if the model had been trained on clean data. In particular, *forward correction* uses corrected loss $\mathcal{L}_T(y_i, \hat{y}^i) = \mathcal{L}(y_i, T\hat{y}^i)$.

Theorem 1. *For any loss function where the minimizing leaf value is the weighted mean, the loss value for a given tree structure is invariant to forward loss correction.*

The proof is in Appendix A.2. From this we conclude that, while forward loss correction may change leaf values, it ultimately does not affect the learned tree structure. Therefore forward correction can be applied by simply learning a tree as usual, then replacing the leaf values with $v_\ell \leftarrow T^{-1}v_\ell$, as shown in the proof. Typically, however, the true class is observed the most frequently, so this correction is unlikely to change the plurality class of leaves, and thus unlikely to improve performance.

In addition to forward correction, there is a related method called *backward correction* (Patrini et al., 2017). For decision trees, backward correction reduces to simply fitting to data $\{(x_i, T^{-1}y_i)\}_{i=1}^n$, as shown in the Appendix A.3. Unlike forward correction, this can result in different tree structure.

Symmetric loss functions are not suitable for decision tree growth. Symmetric loss functions have the property that there exists some constant C such that, for any $\hat{y}$, $\sum_{k=1}^{c} \mathcal{L}(e_k, \hat{y}) = C$, where e_k is the indicator (one-hot) vector with 1 at index k. Under certain assumptions, they are theoretically tolerant to label noise and improve performance of models trained on data with noisy labels compared to models trained with conventional loss functions (Charoenphakdee et al., 2019). Here we assume one-hot training labels, that is, that there are no pseudo-labels in the data.

Theorem 2. *For any symmetric, non-negative loss function, there exist loss-minimizing leaf values that are plurality indicators.*

The proof is in Appendix A.4. Here "plurality" is a multi-class generalization of majority that refers to the most frequent class, without implying that its frequency is greater than one half. The assignment of leaf values to a plurality class indicator poses challenges to tree growth because, particularly with imbalanced data, it frequently occurs that all potential splits lead to both children nodes having the same leaf value, resulting in zero gain. Thus, symmetric loss functions are not suitable for decision tree growth.

EMPIRICAL ANALYSIS

We evaluate the impact of these types of loss correction on the performance of decision tree-based models. We include decision trees, random forests, and ExtraTrees using implementations from the popular scikit-learn library. We use six benchmark data sets from the OpenML data set repository (Vanschoren et al., 2014), outlined in Appendix Table 1. We add a type of label noise called Noise Completely At Random (Frénay & Kabán, 2014), where each sample has a probability of η to be flipped to another label uniformly at random. We use η from 0% to 40% in increments of 10%.

Figure 1 shows the results on "wine" data set. The results of complete six data sets are included in Appendix Figure 2, Figure 3, Figure 4. Overall, there is a lack of evidence that either type of loss correction improves performance of trees under label noise, confirming our theoretical findings.

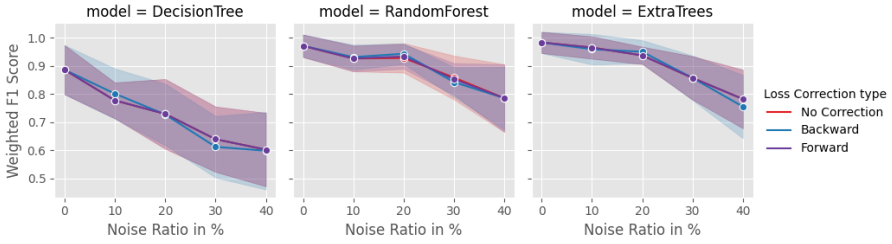

Figure 1: Performance of Decision Tree, Extra Trees and Random Forest models on "wine" data set. We show forward and backward loss-corrected models as well models without loss correction measured by the weighted F1 score. Reported scores are the averages of ten fold cross-validation ploted with standard deviation.

CONCLUSION

Based on our findings, we can conclude that popular techniques of loss design employed in deep learning algorithms to robustly learn models in the presence of label noise are not applicable to decision tree induction. As a result, there is a need for alternative methods that cater specifically to robust learning of decision trees.

ACKNOWLEDGEMENTS

This work was partially funded by the Defense Advanced Research Projects Agency under award FA8750-17-2-0130.

URM STATEMENT

The authors acknowledge that the paper meets the URM criteria of the ICLR 2024 Tiny Papers Track.

REFERENCES

Nontawat Charoenphakdee, Jongyeong Lee, and Masashi Sugiyama. On symmetric losses for learning from corrupted labels. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 961–970. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/charoenphakdee19a.html.

Benoît Frénay and Ata Kabán. A comprehensive introduction to label noise. In *The European Symposium on Artificial Neural Networks*, 2014. URL https://api.semanticscholar.org/CorpusID:17187125.

Aritra Ghosh, Naresh Manwani, and P. S. Sastry. On the Robustness of Decision Tree Learning Under Label Noise. In Jinho Kim, Kyuseok Shim, Longbing Cao, Jae-Gil Lee, Xuemin Lin, and Yang-Sae Moon (eds.), *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pp. 685–697, Cham, 2017. Springer International Publishing. ISBN 978-3-319-57454-7. doi: 10.1007/978-3-319-57454-7_53.

Bo Han, Quanming Yao, Tongliang Liu, Gang Niu, Ivor W. Tsang, James T. Kwok, and Masashi Sugiyama. A Survey of Label-noise Representation Learning: Past, Present and Future, February 2021. URL http://arxiv.org/abs/2011.04406. arXiv:2011.04406 [cs].

Qiguang Miao, Ying Cao, Ge Xia, Maoguo Gong, Jiachen Liu, and Jianfeng Song. Rboost: Label noise-robust boosting algorithm based on a nonconvex loss function and the numerically stable base learners. *IEEE Transactions on Neural Networks and Learning Systems*, 27(11):2216–2228, 2016. doi: 10.1109/TNNLS.2015.2475750.

Géraldin Nanfack, Paul Temple, and Benoît Frénay. Constraint Enforcement on Decision Trees: A Survey. *ACM Computing Surveys*, 54(10s):1–36, January 2022. ISSN 0360-0300, 1557-7341. doi: 10.1145/3506734. URL https://dl.acm.org/doi/10.1145/3506734.

Curtis G. Northcutt, Anish Athalye, and Jonas Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. In *Proceedings of the 35th Conference on Neural Information Processing Systems Track on Datasets and Benchmarks*, December 2021.

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2233–2241, 2017. doi: 10.1109/CVPR.2017.240.

Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 34(11):8135–8153, 2023. doi: 10.1109/TNNLS.2022.3152527.

Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. OpenML: networked science in machine learning. 15(2):49–60, 2014. ISSN 1931-0145. doi: 10.1145/2641190.2641198. URL https://doi.org/10.1145/2641190.2641198.

Bin-Bin Yang, Wei Gao, and Ming Li. On the Robust Splitting Criterion of Random Forest. In *2019 IEEE International Conference on Data Mining (ICDM)*, pp. 1420–1425, November 2019. doi: 10.1109/ICDM.2019.00184. ISSN: 2374-8486.

Xu Zhou, Pak Lun Kevin Ding, and Baoxin Li. Improving robustness of random forest under label noise. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 950–958, 2019. doi: 10.1109/WACV.2019.00106.

A THEORY AND PROOFS

This section contains proofs and other support of theoretical claims made in the main paper.

A.1 TREE FITTING AS LOSS MINIMIZATION

Decision tree fitting recursively splits the input space into leaves such that each split maximally reduces the impurity of the labels in each leaf, summed over the leaves weighted by the number of samples in each. Here we show that these impurity functions are equivalent to loss functions used in parametric learning. We will use the following identities:

1. The total weight of data at a given leaf is $w_\ell = \sum_{i=1}^n \mu_\ell(x_i)$.
2. A leaf value is the mean label of data belonging to the leaf: $v_\ell = \frac{1}{w_\ell} \sum_{i=1}^n \mu_\ell(x_i) y_i$.
3. The predicted value is the value of the leaf to which the sample belongs: $\hat{y}_i = \sum_\ell \mu_\ell(x_i) v_\ell$.

First we show that Gini gain is equivalent to mean squared error loss. The impurity associated with Gini gain is as follows.

$$
\begin{aligned}
&\frac{1}{n} \sum_\ell w_\ell (1 - v_\ell^\top v_\ell) \\
&\frac{1}{n} \sum_\ell w_\ell (1 - 2 v_\ell^\top v_\ell + v_\ell^\top v_\ell) \\
&= 1 - \frac{2}{n} \sum_\ell w_\ell v_\ell^\top v_\ell + \frac{1}{n} \sum_\ell w_\ell v_\ell^\top v_\ell \\
&= 1 - \frac{2}{n} \sum_\ell w_\ell v_\ell^\top \left(\frac{1}{w_\ell} \sum_{i=1}^n \mu_\ell(x_i) y_i \right) + \frac{1}{n} \sum_\ell \left(\sum_{i=1}^n \mu_\ell(x_i) \right) v_\ell^\top v_\ell \\
&= 1 - \frac{2}{n} \sum_\ell v_\ell^\top \left(\sum_{i=1}^n \mu_\ell(x_i) y_i \right) + \frac{1}{n} \sum_{i=1}^n \sum_\ell \mu_\ell(x_i) v_\ell^\top v_\ell
\end{aligned}
$$

For the rightmost term, recall that, for each i, $\mu_\ell(x_i)$ is 1 for exactly one leaf and zero for all others.

$$
\begin{aligned}
&= 1 - \frac{2}{n} \sum_{i=1}^n \left(\sum_\ell \mu_\ell(x_i) v_\ell \right)^\top y_i + \frac{1}{n} \left(\sum_{i=1}^n \sum_\ell \mu_\ell(x_i) v_\ell \right)^\top \left(\sum_{i=1}^n \sum_\ell \mu_\ell(x_i) v_\ell \right) \\
&= \frac{1}{n} \sum_{i=1}^n 1 - 2 \hat{y}^\top y_i + \hat{y}^\top \hat{y}
\end{aligned}
$$

Since y_i is an indicator, $y_i^\top y_i = 1$.

$$
\begin{aligned}
&= \frac{1}{n} \sum_{i=1}^n y_i^\top y_i + \hat{y}^\top \hat{y} - 2 y^\top \hat{y}_i \\
&= \frac{1}{n} \sum_{i=1}^n \| y - \hat{y} \|^2
\end{aligned}
$$

This is the mean squared error.

Next we show that information gain is equivalent to cross-entropy loss. The impurity associated with information gain is as follows.

$$\begin{aligned}
&\frac{1}{n}\sum_{\ell} w_\ell v_\ell^\top \log v_\ell \\
&= \frac{1}{n}\sum_{\ell} w_\ell \frac{1}{w_\ell}\left(\sum_{i=1}^{n}\mu_\ell(x_i)y_i\right)^\top \log v_\ell \\
&= \frac{1}{n}\sum_{\ell}\sum_{i=1}^{n}\mu_\ell(x_i)y_i^\top \log v_\ell \\
&= \frac{1}{n}\sum_{i=1}^{n} y_i^\top \left(\sum_{\ell}\mu_\ell(x_i)\log v_\ell\right)
\end{aligned}$$

For each i, $\mu_\ell(x_i)$ is 1 for exactly one leaf and zero for all others.

$$\begin{aligned}
&= \frac{1}{n}\sum_{i=1}^{n} y_i^\top \log\left(\sum_{\ell}\mu_\ell(x_i)v_\ell\right) \\
&= \frac{1}{n}\sum_{i=1}^{n} y_i^\top \log \hat{y}
\end{aligned}$$

This is the cross-entropy loss.

A.2 Proof of Theorem 1

Assume the leaf values that minimize the loss are the weighted mean of the training data labels in each leaf. Let v_ℓ denote this weighted mean for leaf ℓ. Forward loss correction uses loss $\mathcal{L}_T(y_i, \hat{y}_i) = \mathcal{L}(y_i, T\hat{y}_i)$; and so the minimizing leaf value $v_\ell^{(T)}$ is

$$\begin{aligned}
v_\ell^{(T)} &= \arg\min_v \sum_{i=1}^{n}\mu_\ell(x_i)\mathcal{L}_T(y_i, v) \\
&= \arg\min_v \sum_{i=1}^{n}\mu_\ell(x_i)\mathcal{L}(y_i, Tv).
\end{aligned}$$

By substitution we have

$$\begin{aligned}
Tv_\ell^{(T)} &= \arg\min_v \sum_{i=1}^{n}\mu_\ell(x_i)\mathcal{L}(y_i, v) \\
&= v_\ell
\end{aligned}$$

so $v_\ell^{(T)} = T^{-1}v_\ell$. Then the total corrected loss is

$$\begin{aligned}
&\sum_{\ell}\sum_{i=1}^{n}\mu_\ell(x_i)\mathcal{L}_T(y_i, v_\ell^{(T)}) \\
&= \sum_{\ell}\sum_{i=1}^{n}\mu_\ell(x_i)\mathcal{L}(y_i, TT^{-1}v_\ell) \\
&= \sum_{\ell}\sum_{i=1}^{n}\mu_\ell(x_i)\mathcal{L}(y_i, v_\ell),
\end{aligned}$$

which is the same as the loss without forward correction.

A.3 BACKWARD LOSS CORRECTION FOR DECISION TREES

Let $\vec{\mathcal{L}} : \mathbb{R}^c \to \mathbb{R}^c$ map a prediction $\hat{y}$ to the loss for each possible observed class, that is, $\vec{\mathcal{L}}(\hat{y}) = (\mathcal{L}(e_k, \hat{y}))_{k=1}^c$. Backward loss correction uses corrected loss $\mathcal{L}_T(y, \hat{y}) = y^\top T^{-\top} \vec{\mathcal{L}}(\hat{y})$ where $T^{-\top}$ is the inverse transpose of T.

For a decision tree, the backward corrected loss is as follows.

$$\begin{aligned}
&\sum_{i=1}^n \mathcal{L}_T(y_i, \hat{y}_i) \\
&= \sum_\ell \sum_{i=1}^n \mu_\ell(x_i) \mathcal{L}_T(y_i, v_\ell) \\
&= \sum_\ell \sum_{i=1}^n \mu_\ell(x_i) y_i^\top T^{-\top} \vec{\mathcal{L}}(v_\ell) \\
&= \sum_\ell \sum_{k=1}^c \left(\sum_{i=1}^n \mu_\ell(x_i) y_i^\top T^{-\top} \right)_k \mathcal{L}(e_k, v_\ell) \\
&= \sum_\ell \sum_{k=1}^c w_k^{(T)} \mathcal{L}(e_k, v_\ell).
\end{aligned}$$

which is the same as uncorrected loss, but with corrected weight

$$w_k^{(T)} = \left(\sum_{i=1}^n \mu_\ell(x_i) y_i^\top T^{-\top} \right)_k$$

as opposed to the uncorrected weights

$$w_k = \sum_{i=1}^n \mu_\ell(x_i) \mathbf{1}\{y_i = e_k\}.$$

Thus, assuming the minimizing value of $\mathcal{L}$ is the weighted mean, then the minimizer of the corrected loss is likewise the weighted mean.

$$\begin{aligned}
v_\ell^{(T)} &= \frac{\sum_{k=1}^c w_k^{(T)} e_k}{\sum_{k=1}^c w_k^{(T)}} \\
&= \frac{\sum_{k=1}^c (w_1^{(T)}, \ldots, w_c^{(T)})}{\sum_{k=1}^c w_k^{(T)}} \\
&= \frac{\sum_{i=1}^n \mu_\ell(x_i) y_i^\top T^{-\top}}{\sum_{i=1}^n \mu_\ell(x_i) \sum_k (y_i T^{-\top})_k}
\end{aligned}$$

Since $y_i^\top T^{-\top}$ represents a probability, it sums to 1.

$$= \frac{\sum_{i=1}^n \mu_\ell(x_i) y_i^\top T^{-\top}}{\sum_{i=1}^n \mu_\ell(x_i)}$$

This is the usual leaf value, but computed with labels y_i changed to $y_i^\top T^{-\top}$. Moreover, plugging this back into the loss, we have the following.

$$\sum_{i=1}^n \mathcal{L}_T(y_i, \hat{y}_i) = \sum_\ell \sum_{i=1}^n \mu_\ell(x_i) y_i^\top T^{-\top} \vec{\mathcal{L}}(v_\ell^{(T)})$$

Assume $y^\top \vec{\mathcal{L}}(\hat{y}) = \mathcal{L}(y, \hat{y})$ (shown later).

$$= \sum_\ell \sum_{i=1}^n \mu_\ell(x_i) \mathcal{L}(y_i^\top T^{-\top}, v_\ell^{(T)})$$

This is the usual loss, but computed with labels y_i changed to $y_i^\top T^{-\top}$.

Since both the corrected loss and corrected leaf values can be computed by simply swapping in the corrected labels, we can learn the corrected tree by simply fitting a tree as usual to the corrected data set $\{(x_i, y_i^\top T^{-\top})\}_{i=1}^n$, written equivalently as $\{(x_i, T^{-1}y_i)\}_{i=1}^n$.

We now show that the assumption $y^\top \vec{\mathcal{L}}(\hat{y}) = \mathcal{L}(y, \hat{y})$ holds for the loss functions corresponding to commonly used decision tree impurities.

For mean squared error, corresponding to Gini gain, we have

$$\begin{aligned} y^\top \vec{\mathcal{L}}(\hat{y}) &= \sum_{k=1}^{c} y_k \|e_k - \hat{y}\|^2 \\ &= \sum_{k=1}^{c} y_k (1 + \hat{y}^\top \hat{y} - 2e_k^\top \hat{y}) \end{aligned}$$

Recall that y is an indicator, so $\sum_{k=1}^c y_k = 1$ and $y^\top y = 1$.

$$\begin{aligned} &= y^\top y + \hat{y}^\top \hat{y} - 2y^\top \hat{y} \\ &= \|y - \hat{y}\|^2 \\ &= \mathcal{L}(y, \hat{y}). \end{aligned}$$

For cross-entropy loss, corresponding to information gain, we have

$$\begin{aligned} y^\top \vec{\mathcal{L}}(\hat{y}) &= -\sum_{k=1}^{c} y_k (e_k^\top \log \hat{y}) \\ &= -\sum_{k=1}^{c} y_k \log \hat{y}_k \\ &= \mathcal{L}(y, \hat{y}). \end{aligned}$$

A.4 Proof of Theorem 2

Assume the loss function $\mathcal{L}$ is non-negative and symmetric, that is, for any $\hat{y}$, $\sum_{k=1}^c \mathcal{L}(e_k, \hat{y}) = C$. Then loss at a single leaf ℓ is as follows.

$$\sum_{i=1}^{n} \mu_\ell(x_i) \mathcal{L}(y_i, v_\ell)$$

Let $w_k = \sum_i^n \mu_\ell(x_i)\mathbf{1}\{y_i = e_k\}$ denote the weight of class k at leaf ℓ. Without loss of generality, assume $w_1 \geq w_k$ for $k > 1$, that is, label 1 is a plurality label.

$$
\begin{aligned}
&= \sum_{k=1}^{c} w_k \mathcal{L}(e_k, v_\ell) \\
&= w_1 \mathcal{L}(e_1, v_\ell) + \sum_{k=2}^{c} (w_k + (w_1 - w_1)) \mathcal{L}(e_k, v_\ell) \\
&= w_1 \sum_{k=1}^{c} \mathcal{L}(e_k, v_\ell) + \sum_{k=2}^{c} (w_1 - w_k) \mathcal{L}(e_k, v_\ell) \\
&= w_1 C + \sum_{k=2}^{c} (w_1 - w_k) \mathcal{L}(e_k, v_\ell) \ \ (\mathcal{L} \text{ is symmetric}) \\
&\geq w_1 C \ \ (\mathcal{L} \text{ is non-negative \& } w_1 \geq w_k) \\
&= w_1 \sum_{k=1}^{c} \mathcal{L}(e_k, e_1) \ \ (\mathcal{L} \text{ is symmetric}) \\
&\geq \sum_{k=1}^{c} w_k \mathcal{L}(e_k, e_1) \ \ (w_1 \geq w_k) \\
&= \sum_{i=1}^{n} \mu_\ell(x_i) \mathcal{L}(y_i, e_1)
\end{aligned}
$$

This is the loss with $v_\ell = e_1$. Therefore the leaf value $v_\ell = e_1$, the indicator of a plurality label, is a minimizer of the loss.

B DATA SETS USED IN EXPERIMENTS

data set	OpenML ID	n	m	c
iris	61	150	4	3
optdigits	28	5620	64	10
pendigits	32	10992	16	10
vehicle	54	846	18	4
wine	187	178	13	3
wdbc	1510	569	30	2

Table 1: Description of the data sets used in experimental study.

C EXPERIMENTAL RESULTS

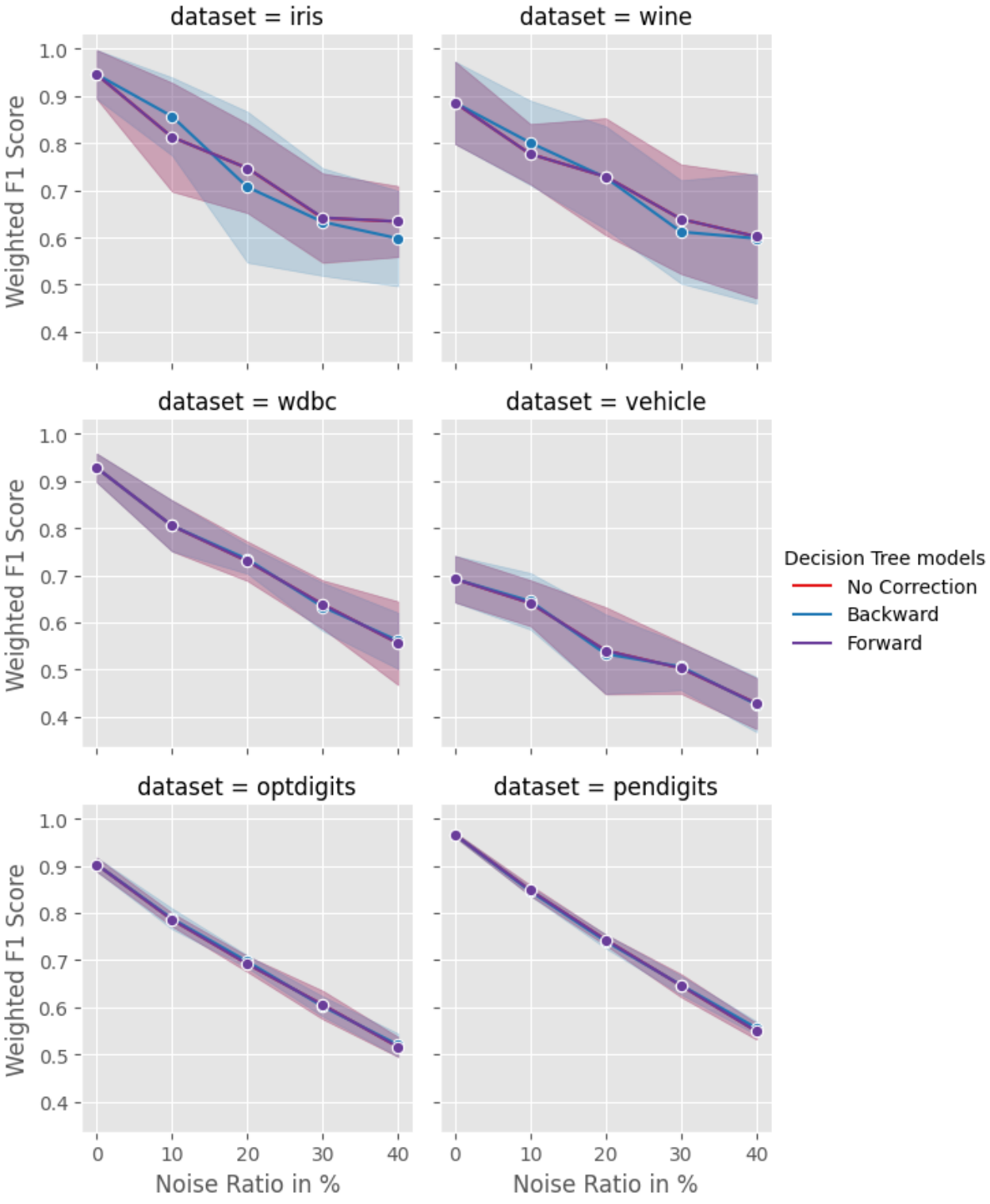

Figure 2: Performance of Decision Tree forward and backward loss-corrected models as well models without loss correction measured by the weighted F1 score on six benchmarking data sets. Reported scores are the averages of ten fold cross-validation ploted with standard deviation.

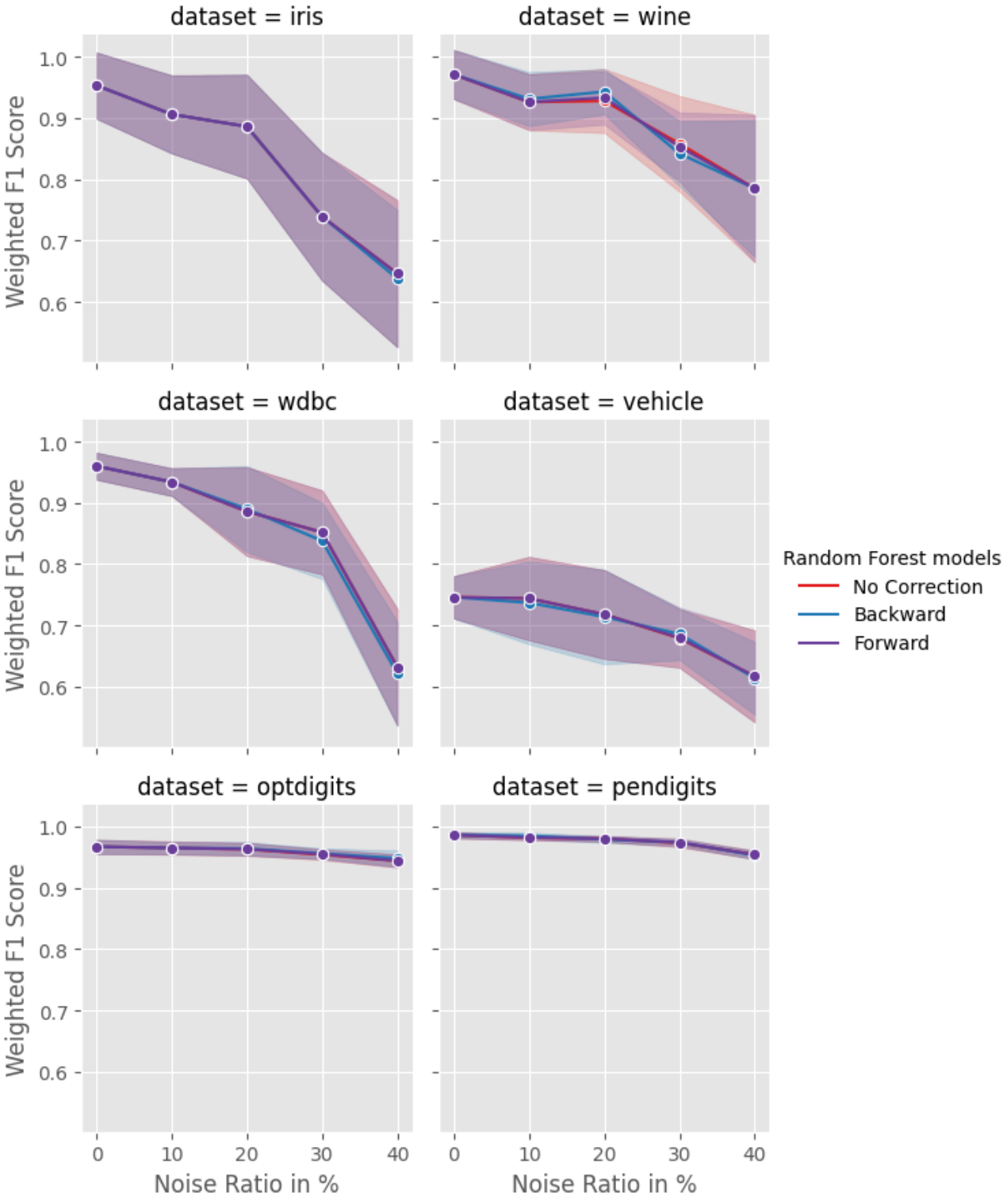

Figure 3: Performance of Random Forest forward and backward loss-corrected models as well models without loss correction measured by the weighted F1 score on six benchmarking data sets. Reported scores are the averages of ten fold cross-validation ploted with standard deviation.

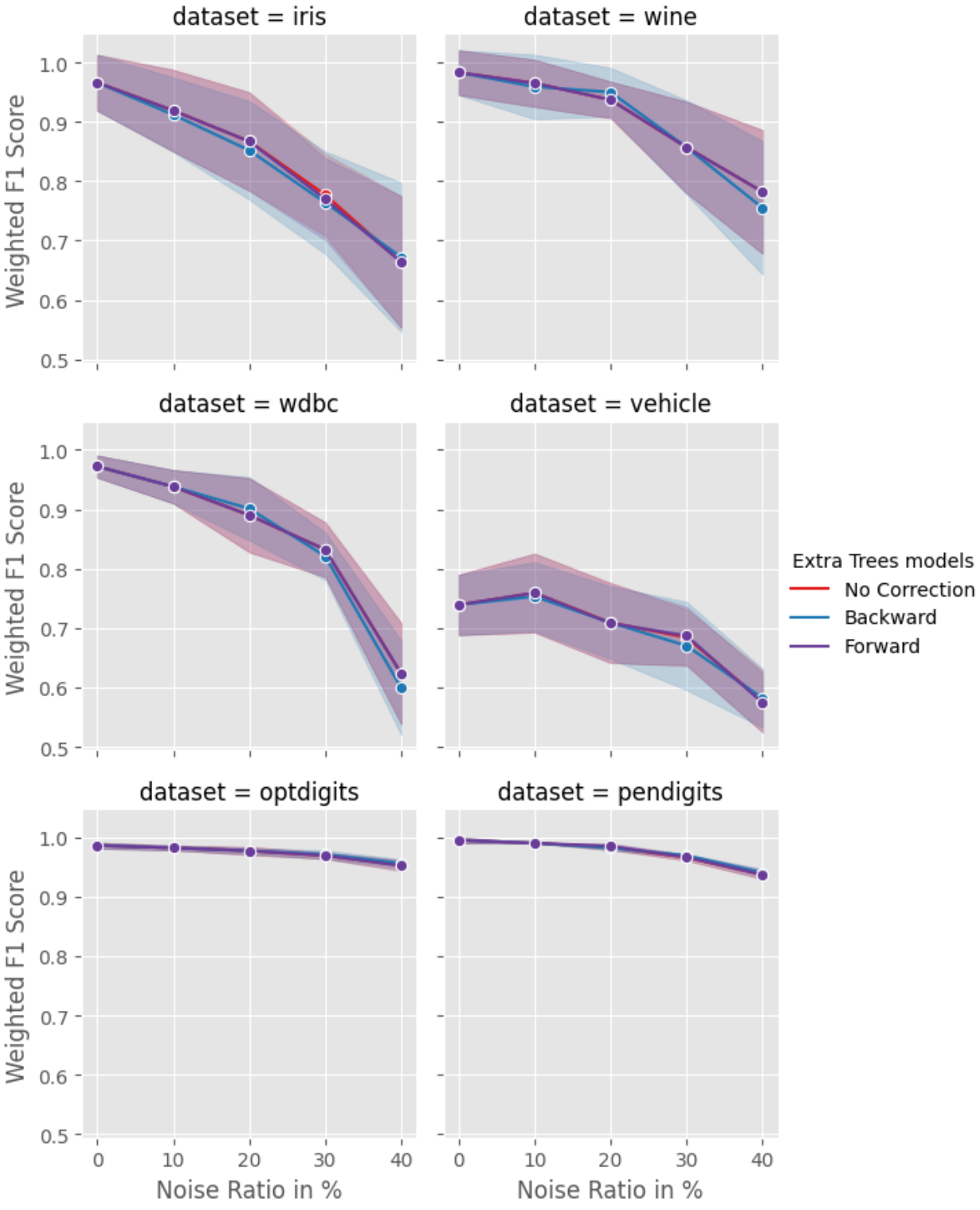

Figure 4: Performance of Extra Trees forward and backward loss-corrected models as well models without loss correction measured by the weighted F1 score on six benchmarking data sets. Reported scores are the averages of ten fold cross-validation ploted with standard deviation.

Noise Ratio	Model	Data set / Loss Correction Type	iris	optdigits	pendigits
0	DecisionTree	Backward	0.946 ± 0.052	0.903 ± 0.015	0.966 ± 0.005
0	DecisionTree	Forward	0.946 ± 0.052	0.903 ± 0.015	0.966 ± 0.005
0	DecisionTree	No Correction	0.946 ± 0.052	0.903 ± 0.015	0.966 ± 0.005
0	ExtraTrees	Backward	0.966 ± 0.047	0.986 ± 0.005	0.995 ± 0.005
0	ExtraTrees	Forward	0.966 ± 0.047	0.986 ± 0.005	0.995 ± 0.005
0	ExtraTrees	No Correction	0.966 ± 0.047	0.986 ± 0.005	0.995 ± 0.005
0	RandomForest	Backward	0.953 ± 0.054	0.967 ± 0.012	0.986 ± 0.005
0	RandomForest	Forward	0.953 ± 0.054	0.967 ± 0.012	0.986 ± 0.005
0	RandomForest	No Correction	0.953 ± 0.054	0.967 ± 0.012	0.986 ± 0.005
10	DecisionTree	Backward	0.857 ± 0.083	0.788 ± 0.022	0.846 ± 0.010
10	DecisionTree	Forward	0.813 ± 0.115	0.786 ± 0.014	0.848 ± 0.011
10	DecisionTree	No Correction	0.813 ± 0.115	0.786 ± 0.014	0.848 ± 0.011
10	ExtraTrees	Backward	0.912 ± 0.063	0.982 ± 0.004	0.990 ± 0.000
10	ExtraTrees	Forward	0.919 ± 0.069	0.982 ± 0.004	0.990 ± 0.000
10	ExtraTrees	No Correction	0.919 ± 0.069	0.982 ± 0.004	0.990 ± 0.000
10	RandomForest	Backward	0.906 ± 0.064	0.965 ± 0.010	0.985 ± 0.005
10	RandomForest	Forward	0.906 ± 0.064	0.965 ± 0.011	0.982 ± 0.004
10	RandomForest	No Correction	0.906 ± 0.064	0.965 ± 0.011	0.982 ± 0.004
20	DecisionTree	Backward	0.707 ± 0.160	0.697 ± 0.013	0.740 ± 0.015
20	DecisionTree	Forward	0.747 ± 0.095	0.692 ± 0.017	0.743 ± 0.012
20	DecisionTree	No Correction	0.747 ± 0.095	0.692 ± 0.017	0.743 ± 0.012
20	ExtraTrees	Backward	0.852 ± 0.083	0.977 ± 0.005	0.982 ± 0.004
20	ExtraTrees	Forward	0.867 ± 0.083	0.977 ± 0.007	0.985 ± 0.005
20	ExtraTrees	No Correction	0.867 ± 0.083	0.977 ± 0.007	0.983 ± 0.005
20	RandomForest	Backward	0.886 ± 0.085	0.964 ± 0.011	0.979 ± 0.006
20	RandomForest	Forward	0.886 ± 0.085	0.963 ± 0.011	0.980 ± 0.005
20	RandomForest	No Correction	0.886 ± 0.085	0.963 ± 0.011	0.980 ± 0.005
30	DecisionTree	Backward	0.633 ± 0.114	0.602 ± 0.021	0.647 ± 0.020
30	DecisionTree	Forward	0.642 ± 0.094	0.605 ± 0.030	0.646 ± 0.025
30	DecisionTree	No Correction	0.641 ± 0.095	0.605 ± 0.030	0.646 ± 0.025
30	ExtraTrees	Backward	0.764 ± 0.086	0.971 ± 0.007	0.969 ± 0.003
30	ExtraTrees	Forward	0.770 ± 0.069	0.969 ± 0.006	0.966 ± 0.005
30	ExtraTrees	No Correction	0.777 ± 0.069	0.969 ± 0.006	0.966 ± 0.005
30	RandomForest	Backward	0.739 ± 0.105	0.956 ± 0.008	0.974 ± 0.005
30	RandomForest	Forward	0.739 ± 0.105	0.955 ± 0.008	0.974 ± 0.007
30	RandomForest	No Correction	0.739 ± 0.105	0.954 ± 0.008	0.973 ± 0.007
40	DecisionTree	Backward	0.598 ± 0.101	0.520 ± 0.024	0.556 ± 0.014
40	DecisionTree	Forward	0.634 ± 0.075	0.516 ± 0.021	0.549 ± 0.017
40	DecisionTree	No Correction	0.634 ± 0.075	0.516 ± 0.021	0.549 ± 0.017
40	ExtraTrees	Backward	0.672 ± 0.054	0.956 ± 0.007	0.940 ± 0.007
40	ExtraTrees	Forward	0.664 ± 0.047	0.952 ± 0.009	0.937 ± 0.008
40	ExtraTrees	No Correction	0.664 ± 0.047	0.953 ± 0.008	0.937 ± 0.007
40	RandomForest	Backward	0.638 ± 0.052	0.948 ± 0.013	0.953 ± 0.007
40	RandomForest	Forward	0.646 ± 0.049	0.944 ± 0.011	0.954 ± 0.007
40	RandomForest	No Correction	0.646 ± 0.049	0.944 ± 0.011	0.954 ± 0.007

Table 2: Detailed performance of forward, backward loss-corrected models and models without loss correction measured by the weighted F1 score on ”iris”, ”optdigits” and ”pendigits” data sets. Scores are the averages of ten fold cross-validation reported with standard deviation.

Noise Ratio	Model	Data set / Loss Correction Type	vehicle	wdbc	wine
0	DecisionTree	Backward	0.692 ± 0.049	0.928 ± 0.030	0.886 ± 0.087
0	DecisionTree	Forward	0.692 ± 0.049	0.928 ± 0.030	0.886 ± 0.087
0	DecisionTree	No Correction	0.692 ± 0.049	0.928 ± 0.030	0.886 ± 0.087
0	ExtraTrees	Backward	0.739 ± 0.051	0.972 ± 0.019	0.983 ± 0.038
0	ExtraTrees	Forward	0.739 ± 0.051	0.972 ± 0.019	0.983 ± 0.038
0	ExtraTrees	No Correction	0.739 ± 0.051	0.972 ± 0.019	0.983 ± 0.038
0	RandomForest	Backward	0.746 ± 0.035	0.960 ± 0.022	0.971 ± 0.040
0	RandomForest	Forward	0.746 ± 0.035	0.960 ± 0.022	0.971 ± 0.040
0	RandomForest	No Correction	0.746 ± 0.035	0.960 ± 0.022	0.971 ± 0.040
10	DecisionTree	Backward	0.645 ± 0.060	0.805 ± 0.054	0.801 ± 0.090
10	DecisionTree	Forward	0.641 ± 0.049	0.805 ± 0.054	0.777 ± 0.064
10	DecisionTree	No Correction	0.641 ± 0.049	0.805 ± 0.054	0.777 ± 0.064
10	ExtraTrees	Backward	0.753 ± 0.058	0.938 ± 0.028	0.959 ± 0.054
10	ExtraTrees	Forward	0.759 ± 0.067	0.938 ± 0.028	0.965 ± 0.040
10	ExtraTrees	No Correction	0.759 ± 0.067	0.938 ± 0.028	0.965 ± 0.040
10	RandomForest	Backward	0.737 ± 0.068	0.934 ± 0.023	0.931 ± 0.044
10	RandomForest	Forward	0.744 ± 0.068	0.934 ± 0.023	0.926 ± 0.046
10	RandomForest	No Correction	0.744 ± 0.068	0.934 ± 0.023	0.926 ± 0.046
20	DecisionTree	Backward	0.533 ± 0.084	0.734 ± 0.031	0.727 ± 0.109
20	DecisionTree	Forward	0.540 ± 0.092	0.730 ± 0.042	0.729 ± 0.124
20	DecisionTree	No Correction	0.540 ± 0.092	0.730 ± 0.042	0.729 ± 0.124
20	ExtraTrees	Backward	0.709 ± 0.061	0.901 ± 0.053	0.950 ± 0.041
20	ExtraTrees	Forward	0.708 ± 0.067	0.890 ± 0.062	0.937 ± 0.031
20	ExtraTrees	No Correction	0.709 ± 0.067	0.890 ± 0.062	0.937 ± 0.031
20	RandomForest	Backward	0.714 ± 0.077	0.890 ± 0.070	0.943 ± 0.037
20	RandomForest	Forward	0.718 ± 0.073	0.886 ± 0.073	0.933 ± 0.043
20	RandomForest	No Correction	0.718 ± 0.073	0.886 ± 0.073	0.928 ± 0.052
30	DecisionTree	Backward	0.506 ± 0.050	0.633 ± 0.051	0.612 ± 0.110
30	DecisionTree	Forward	0.503 ± 0.054	0.638 ± 0.051	0.639 ± 0.116
30	DecisionTree	No Correction	0.503 ± 0.054	0.638 ± 0.051	0.639 ± 0.116
30	ExtraTrees	Backward	0.670 ± 0.074	0.821 ± 0.040	0.857 ± 0.079
30	ExtraTrees	Forward	0.687 ± 0.049	0.832 ± 0.046	0.857 ± 0.079
30	ExtraTrees	No Correction	0.684 ± 0.048	0.832 ± 0.046	0.857 ± 0.079
30	RandomForest	Backward	0.686 ± 0.042	0.838 ± 0.062	0.842 ± 0.053
30	RandomForest	Forward	0.680 ± 0.049	0.852 ± 0.069	0.853 ± 0.056
30	RandomForest	No Correction	0.679 ± 0.047	0.852 ± 0.069	0.858 ± 0.078
40	DecisionTree	Backward	0.426 ± 0.059	0.561 ± 0.060	0.598 ± 0.138
40	DecisionTree	Forward	0.428 ± 0.052	0.556 ± 0.089	0.602 ± 0.089
40	DecisionTree	No Correction	0.428 ± 0.052	0.556 ± 0.089	0.602 ± 0.089
40	ExtraTrees	Backward	0.582 ± 0.049	0.599 ± 0.080	0.755 ± 0.112
40	ExtraTrees	Forward	0.574 ± 0.050	0.623 ± 0.083	0.782 ± 0.120
40	ExtraTrees	No Correction	0.577 ± 0.052	0.623 ± 0.083	0.782 ± 0.120
40	RandomForest	Backward	0.614 ± 0.017	0.621 ± 0.018	0.785 ± 0.112
40	RandomForest	Forward	0.617 ± 0.008	0.632 ± 0.010	0.785 ± 0.112
40	RandomForest	No Correction	0.617 ± 0.008	0.632 ± 0.010	0.785 ± 0.112

Table 3: Detailed performance of forward, backward loss-corrected models and models without loss correction measured by the weighted F1 score on ”vehicle”, ”wdbc” and ”wine” data sets. Scores are the averages of ten fold cross-validation reported with standard deviation.