Credit Scoring and Credit Control XIV 2015
Handling Improbable Labels

Lewis Evans (Imperial College London)

Imperial College London

August, 2015

Overview

- We focus on credit risk classification problems: predicting customer defaults using scorecards
- We present a new approach to build better scorecards through data modification. The training data is modified by handling improbable labels (HIL)
- The final scorecard meets regulatory constraints, while the data modification process may use more sophisticated classifiers (e.g. random forest)
- We demonstrate algorithm effectiveness by experimental studies with credit application data

This research is supervised by Professor Niall M. Adams ‡ † and Dr Christoforos Anagnostopoulos †, and funded by the EPRSC.

‡ Heilbronn Institute for Mathematical Research, University of Bristol

† Imperial College, London

Talk Structure

- Overview
- Motivation
- HIL Estimation Algorithms
- Experimental Study
- Conclusion

Motivation (1)

Assertion: improbable labels cause problems for classifiers; handling improbable labels by data modification improves classifier performance

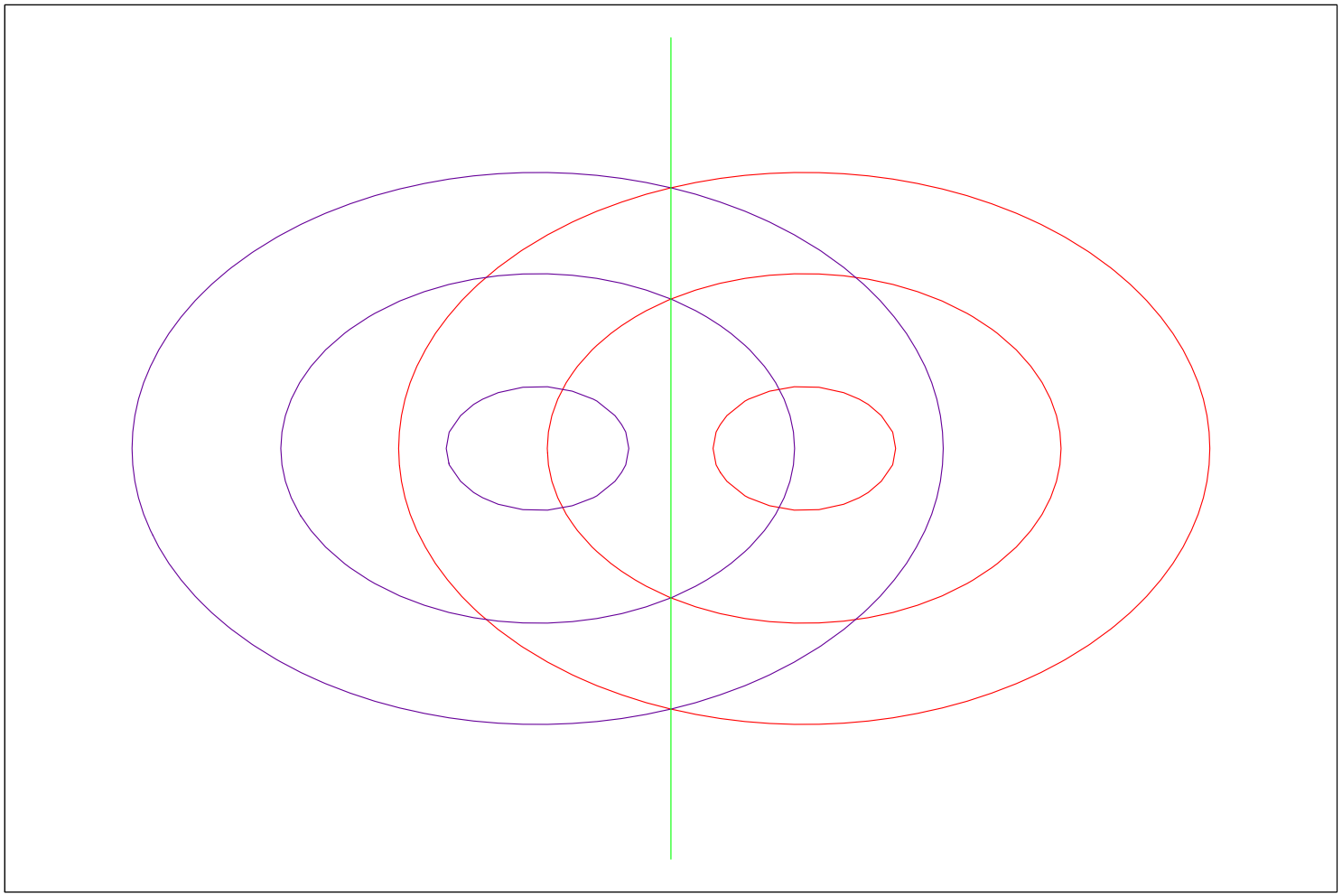
Improbable labels are defined by class probabilities. Suppose a covariate example x has class probability vector $(0.8, 0.2)$, then it is more likely to be class one.

If we see a labelled example (x, c_2), then its label c_2 has low probability:

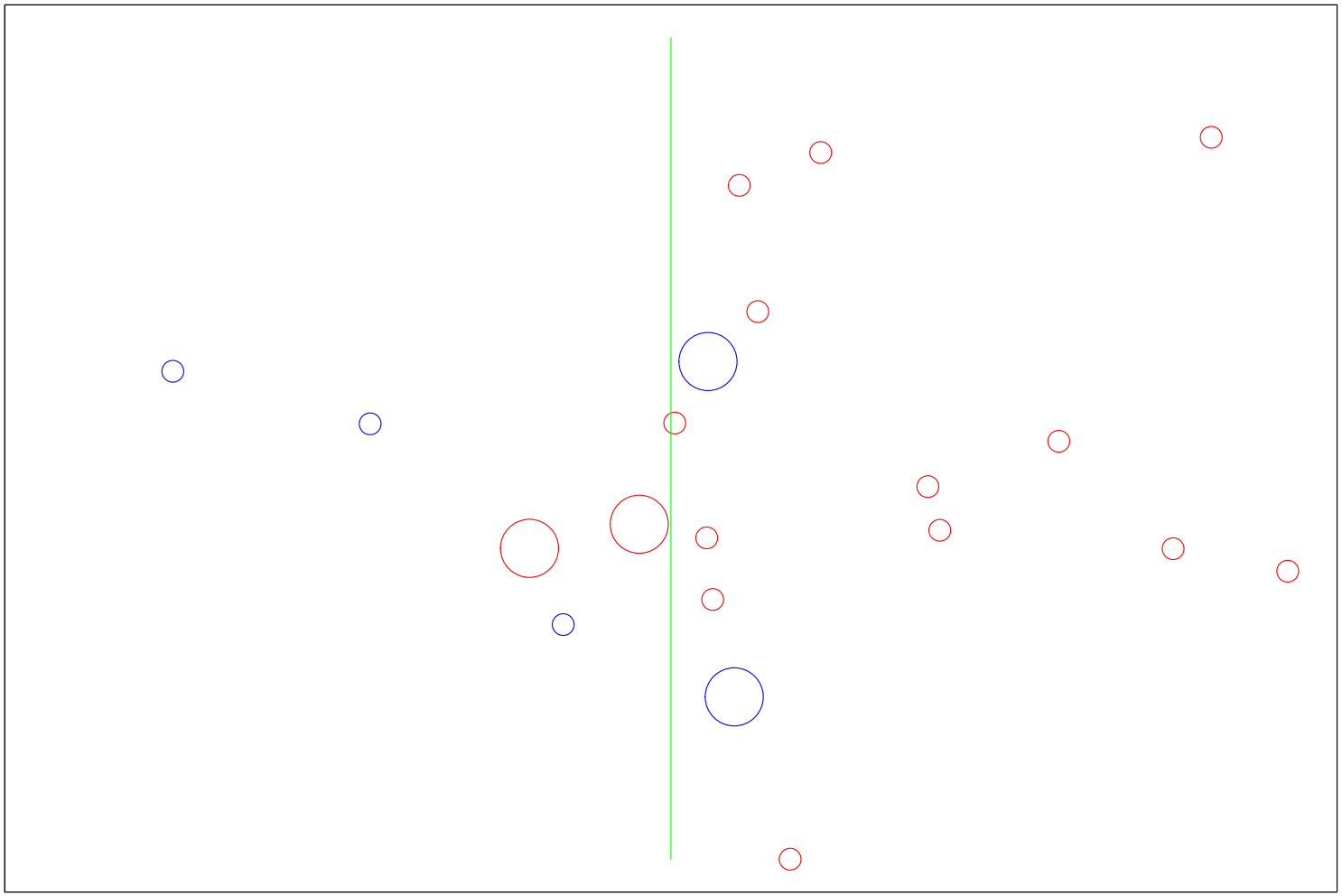
$$p(Y|x) < 0.5.$$

This is how we define improbable labels, as unexpected, low probability labels.

Motivation (2)



Motivation (2)



Motivation (3)

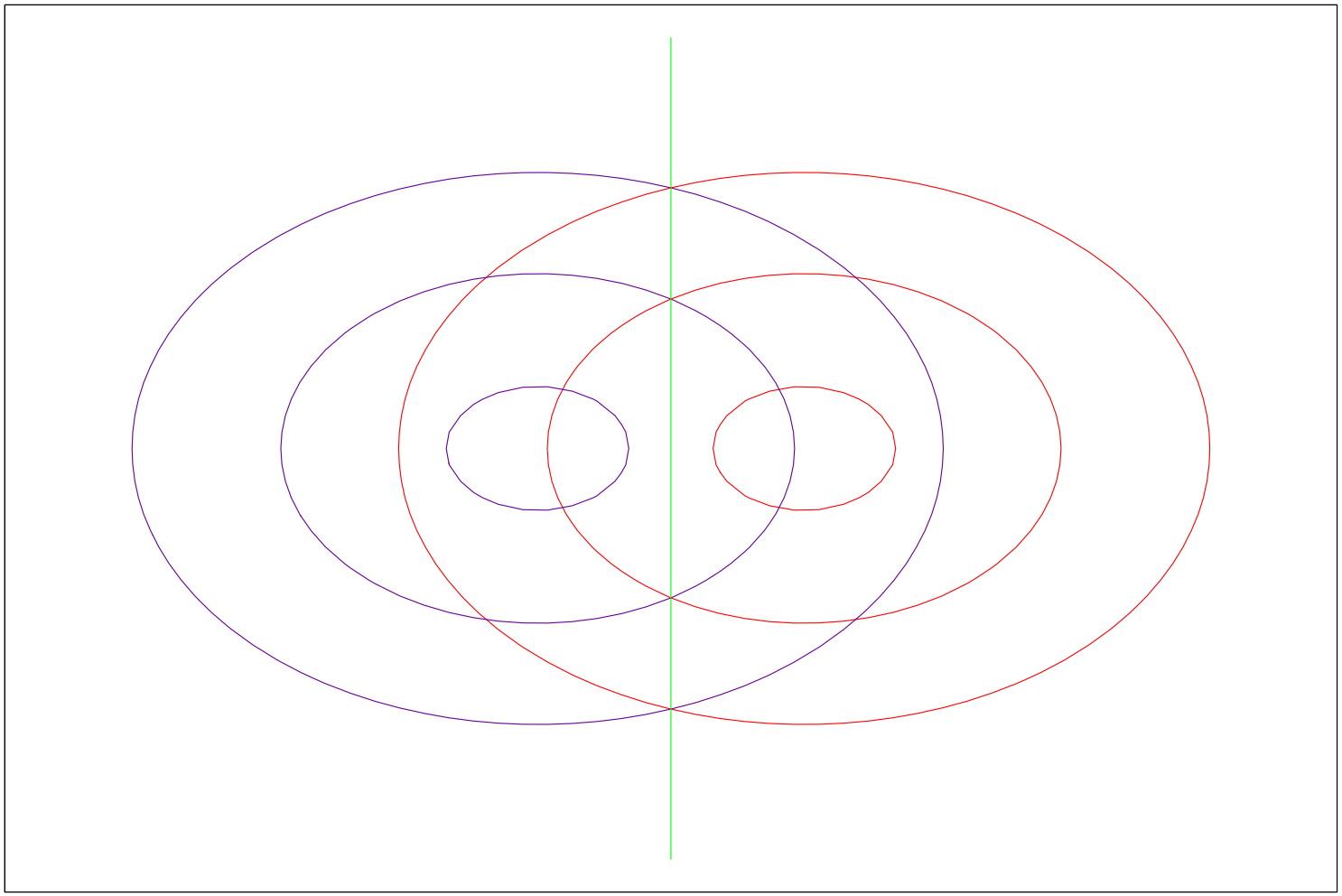
HIL Pruning

- One way to handle the improbable labels is simply to prune those examples, yielding a smaller training dataset
- A new HIL classifier is then trained on this smaller dataset
- The goal is for the HIL classifier to outperform the original classifier
- Both these classifiers are of the same model type
- For example, if the classifier is logistic regression, then the original classifier and the HIL classifier are both examples of logistic regression, trained on different data (same model type, different estimated parameters)

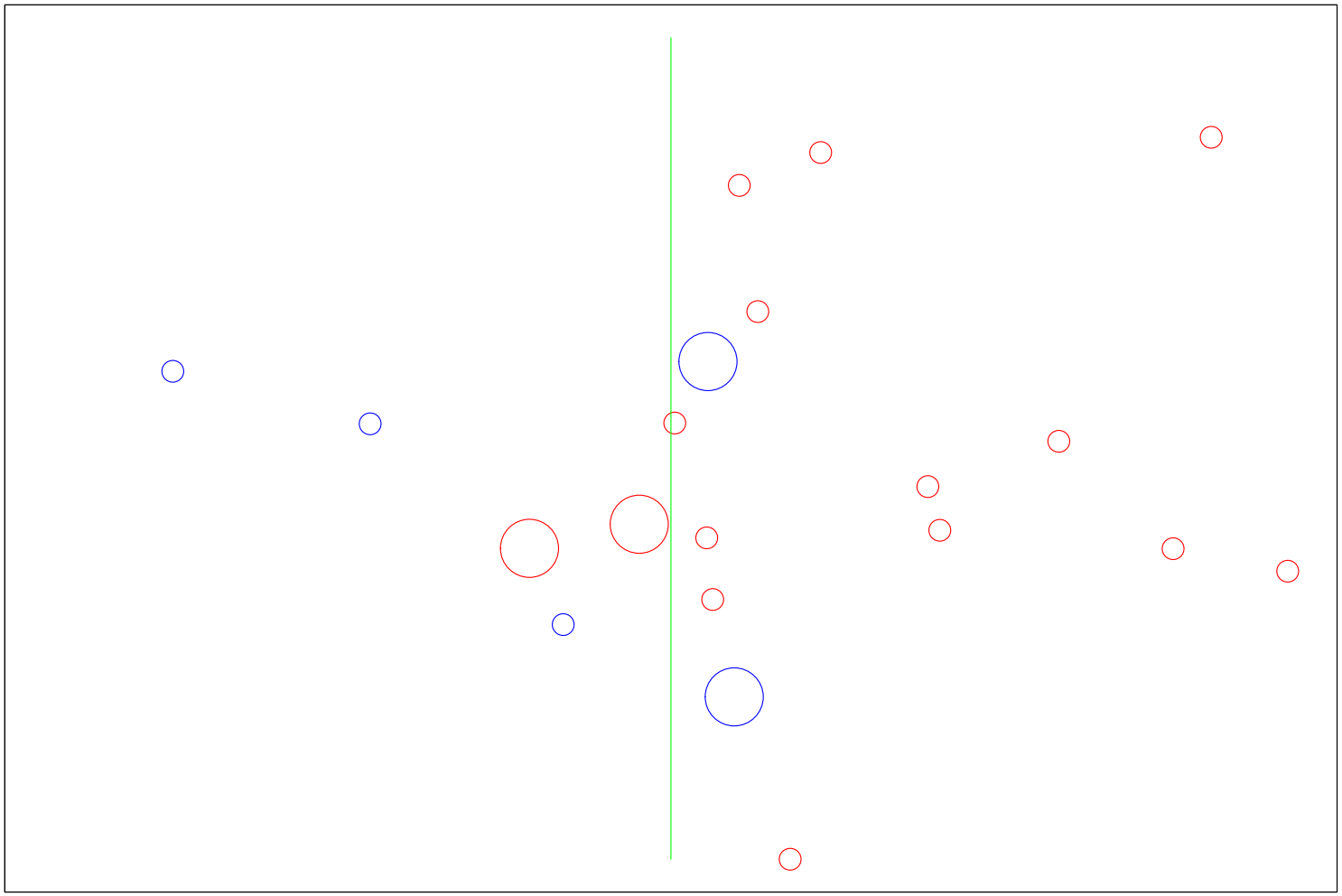
$$\hat{\boldsymbol{\theta}}^{o} = \theta(\mathbf{d}_T)$$

$$\hat{\boldsymbol{\theta}}^{h} = \theta(\mathbf{d}_H)$$

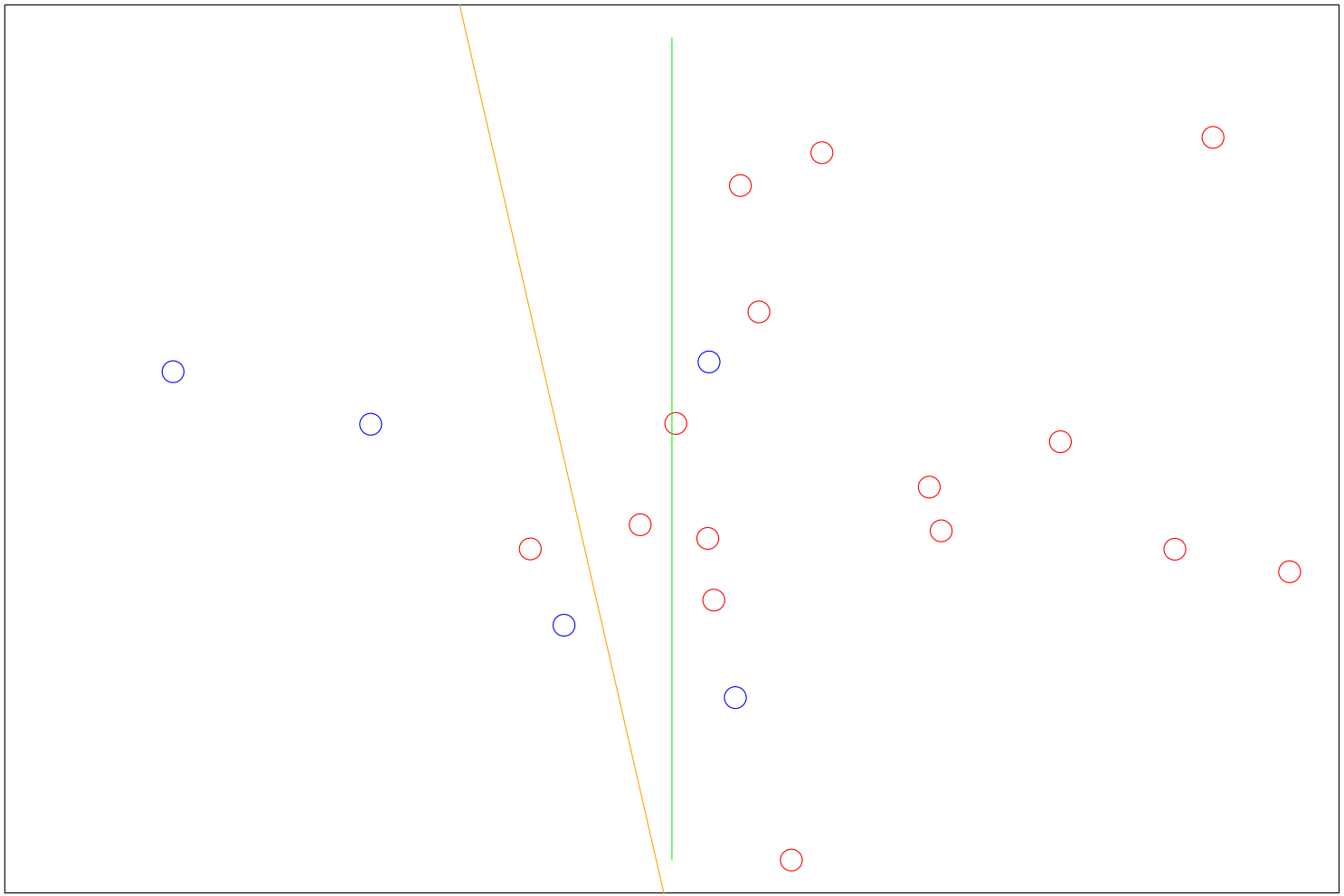
Motivation (4)



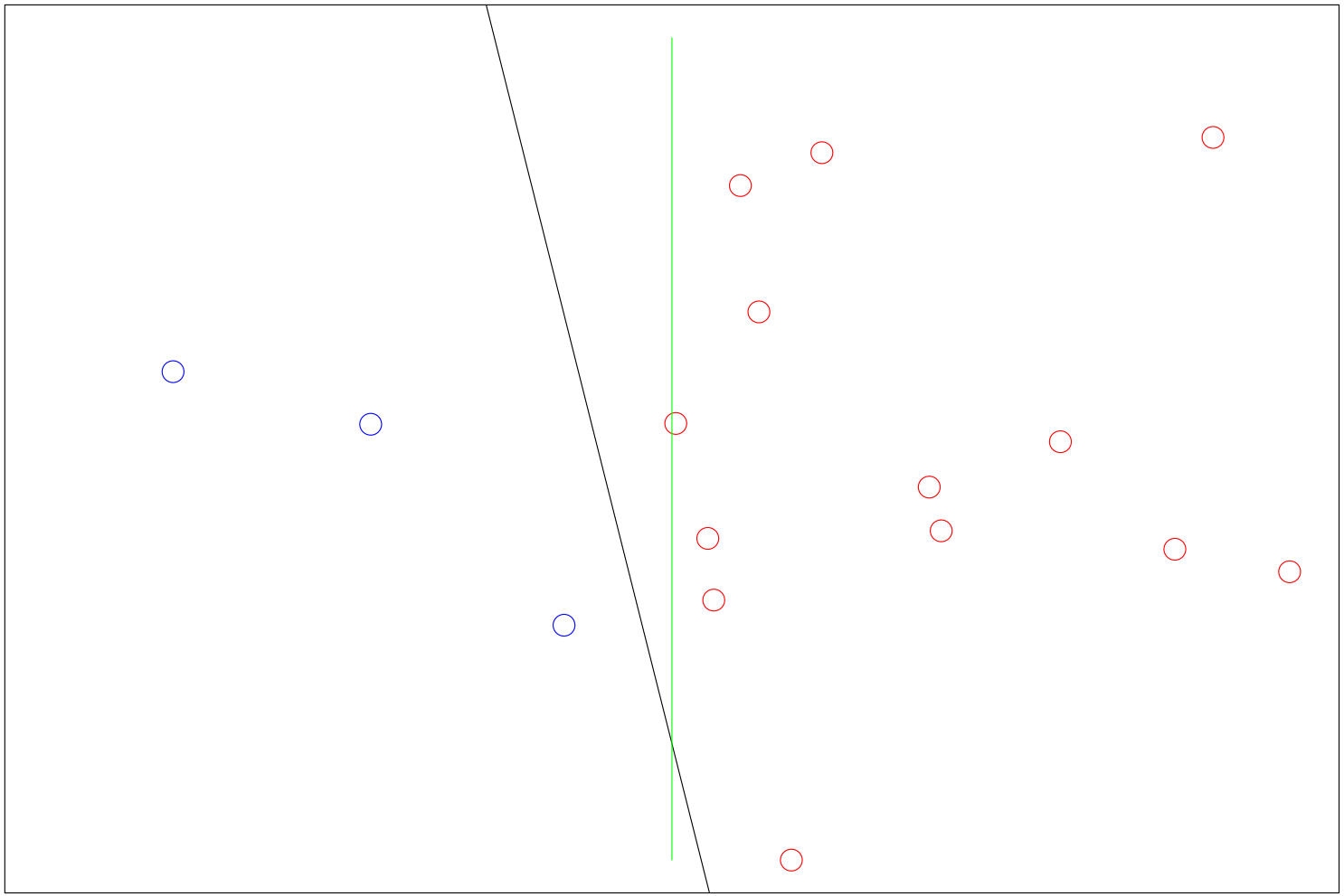
Motivation (4)



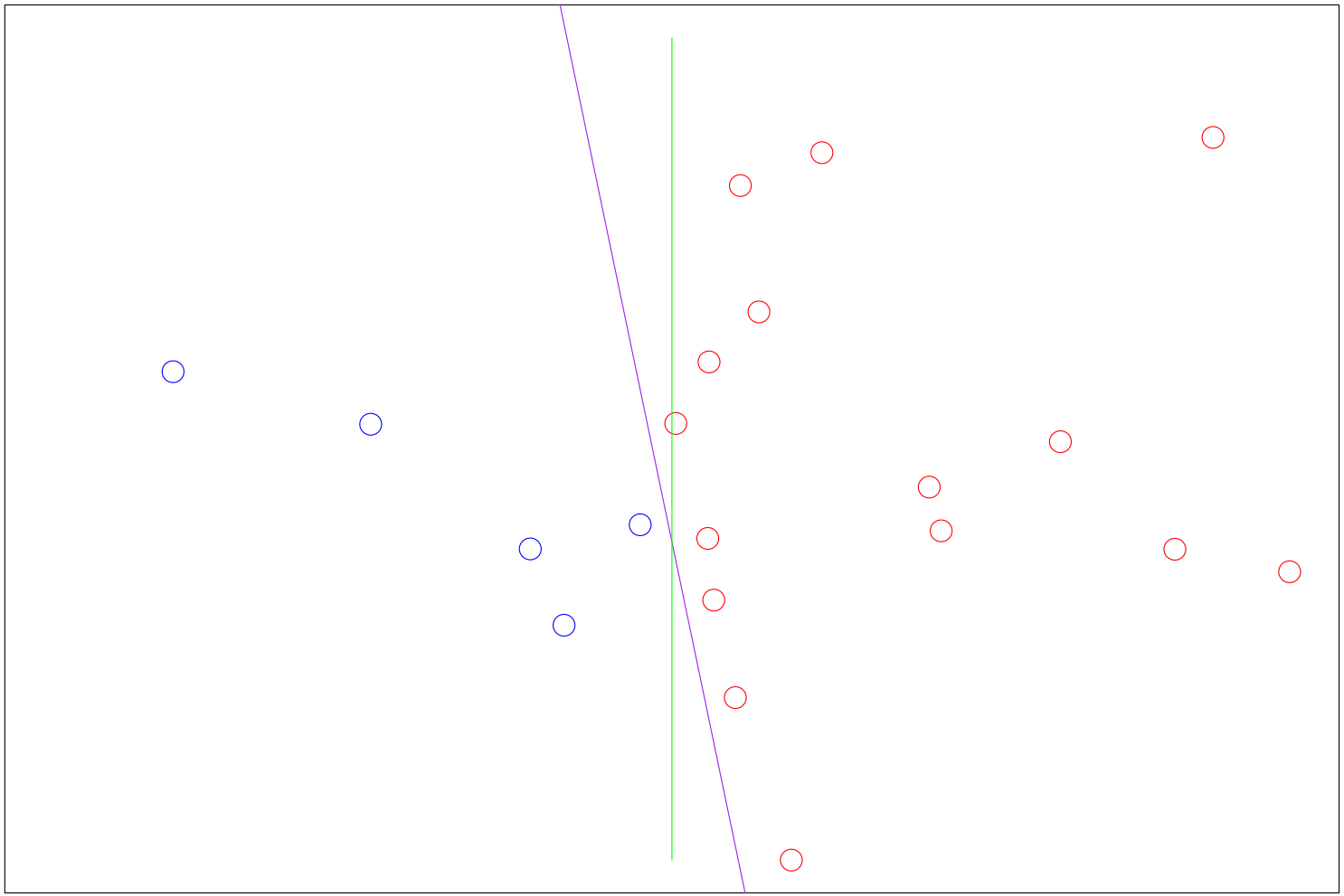
Motivation (4)



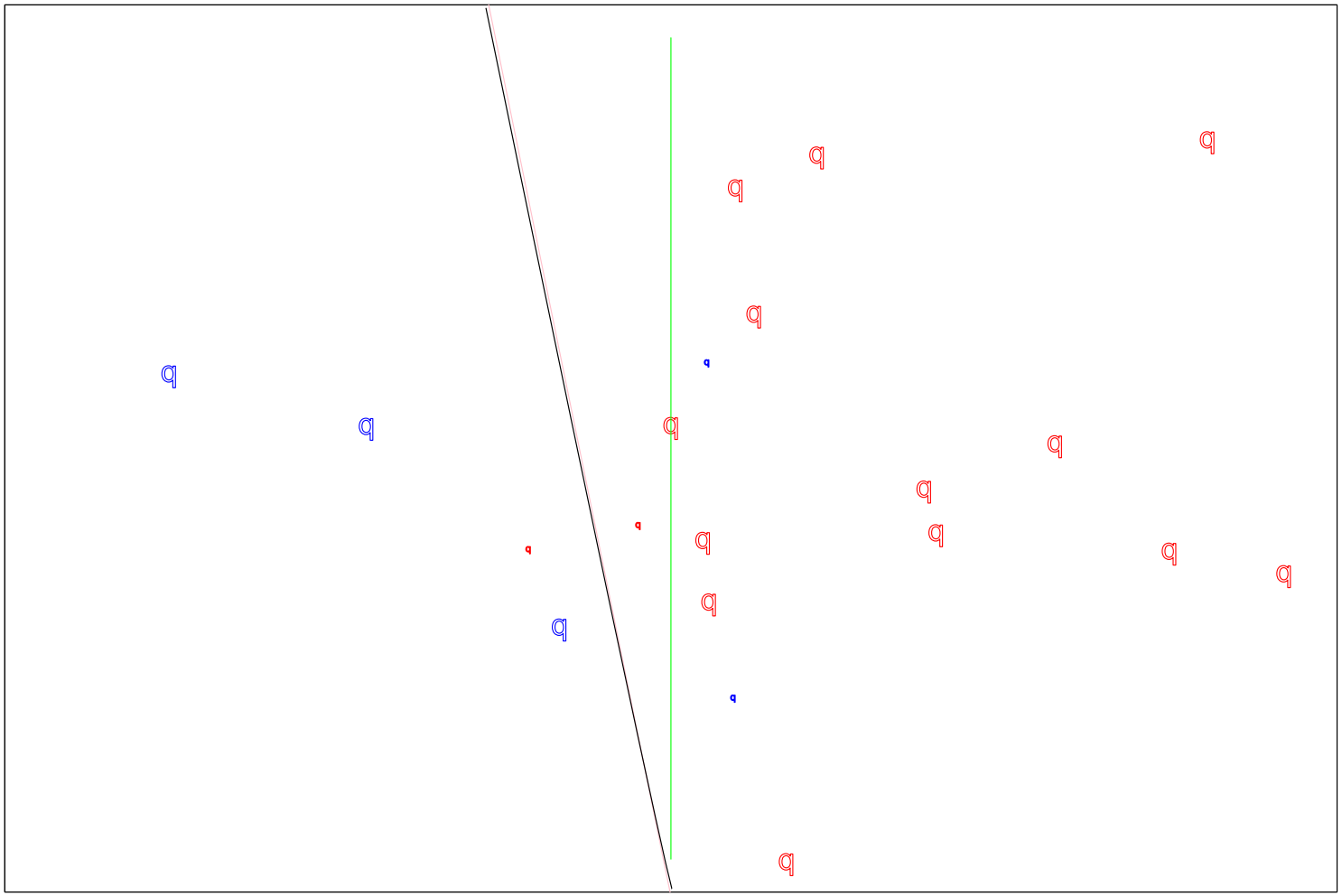
Motivation (4)



Motivation (4)



Motivation (4)



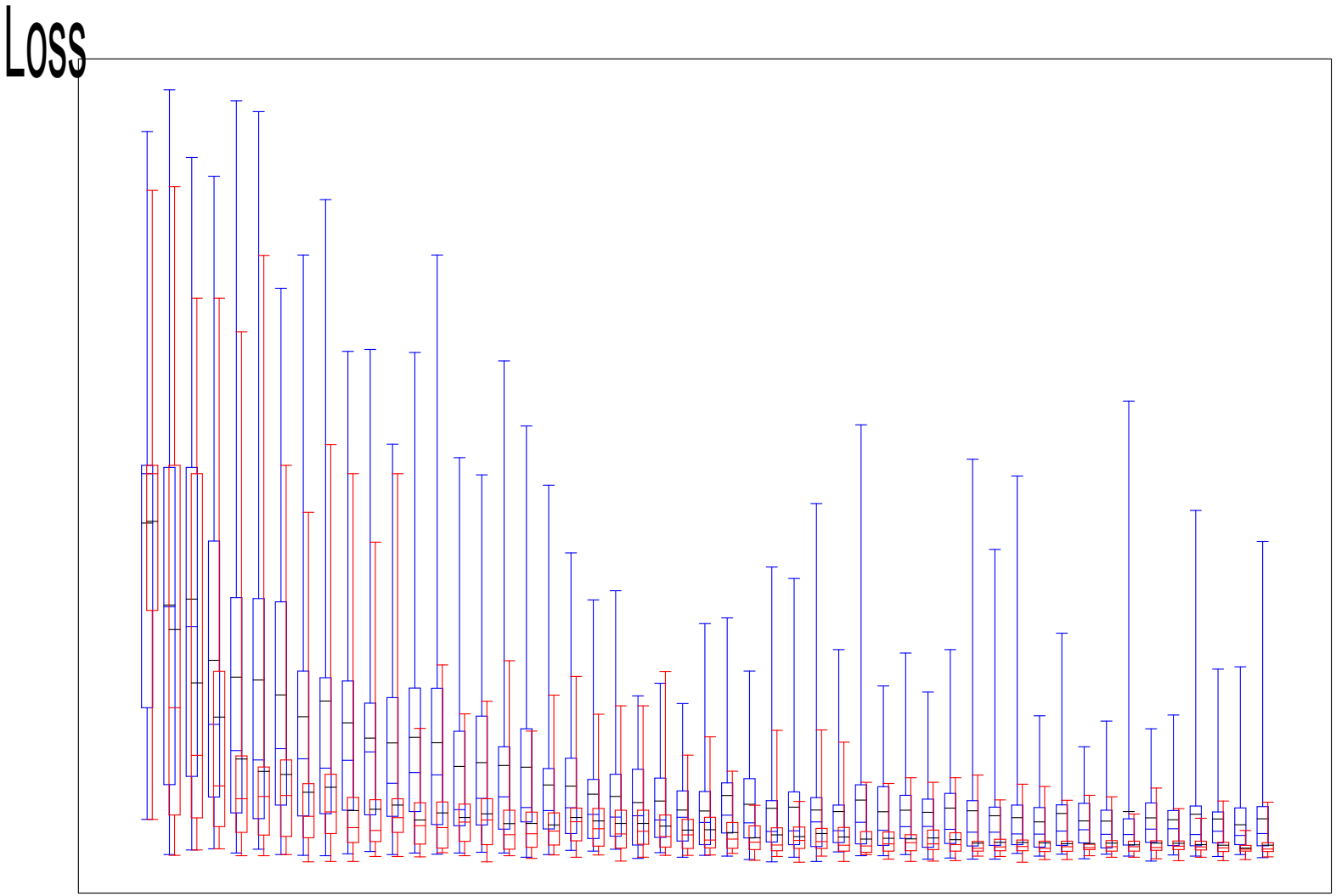
Motivation (5)

We demonstrate the potential performance gain for a simple binary problem (mix of Gaussians), with logistic regression and error rate.

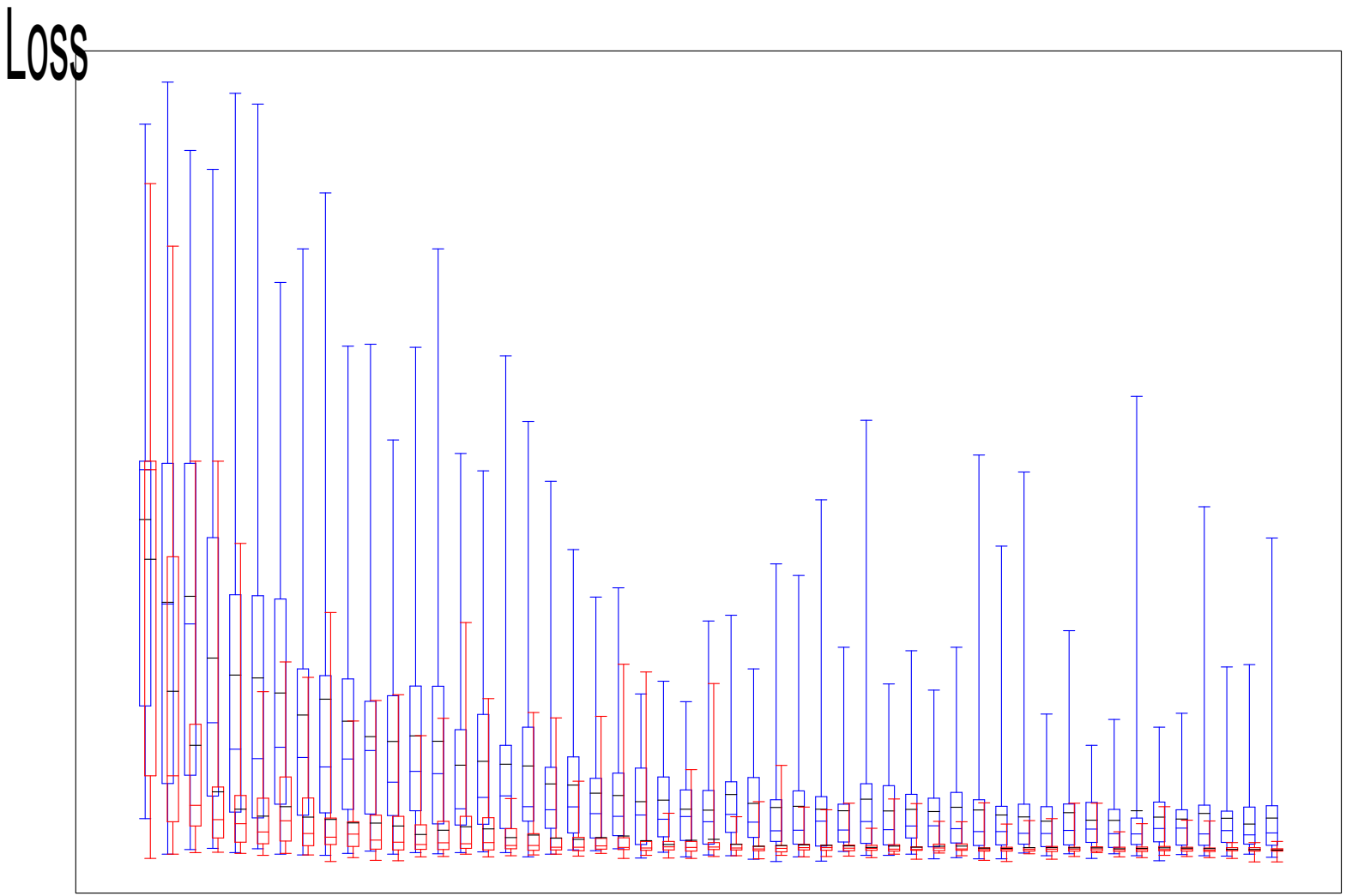
This example uses the true probabilities from the Bayes classifier.

The actions of relabelling and pruning show improved performance over the entire learning curve, whether training data is scarce or abundant.

Motivation (6)



Motivation (6)



HIL Estimation Algorithms (1)

We now change context, to focus on applications with real data.

HIL is a two-step process of estimation and action

- First identify the set of improbable labels. They form an index set Z, which is unknown for real data. We estimate it by $\hat{Z}$, using probability estimates.

$$Z(\mathbf{d}_T) = \{i \in T : y_i \neq y^{MAP}(\mathbf{x}_i)\},$$
$$\hat{Z}(\mathbf{d}_T) = \{i \in T : y_i \neq \hat{y}^{MAP}(\mathbf{x}_i)\}.$$

- Second, take action on $\hat{Z}$: prune, relabel or weight, changing the training data $\mathbf{d}_T$ to $\mathbf{d}_H$.

This two-step process produces a new training dataset $\mathbf{d}_H$, the goal being to reduce classifier loss L

$$L(\theta(\mathbf{d}_H)) < L(\theta(\mathbf{d}_T)),$$

where L is a loss function (e.g. AUC)

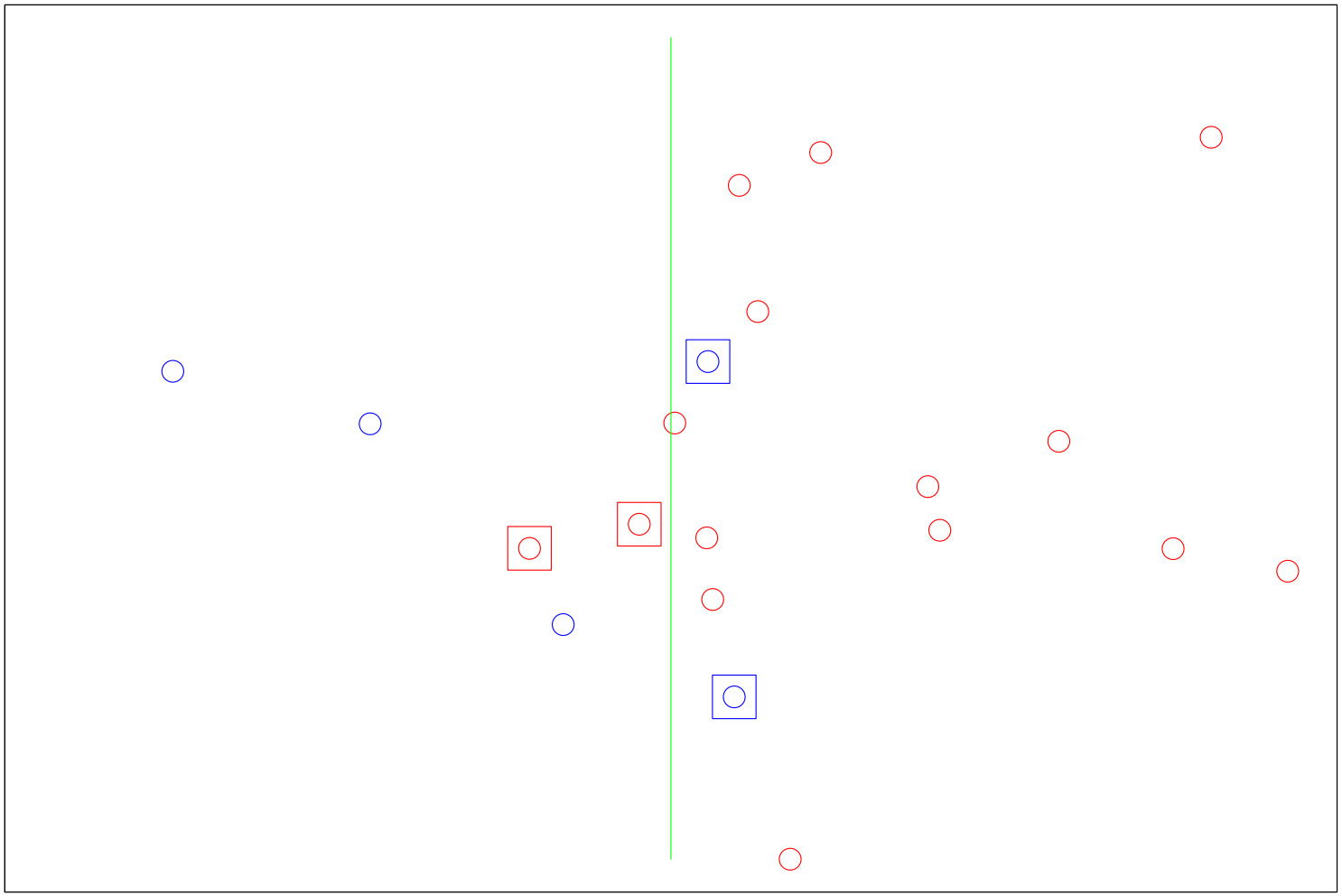
HIL Estimation Algorithms (2)

The earlier example showed an abstract problem, where exact probabilities are known

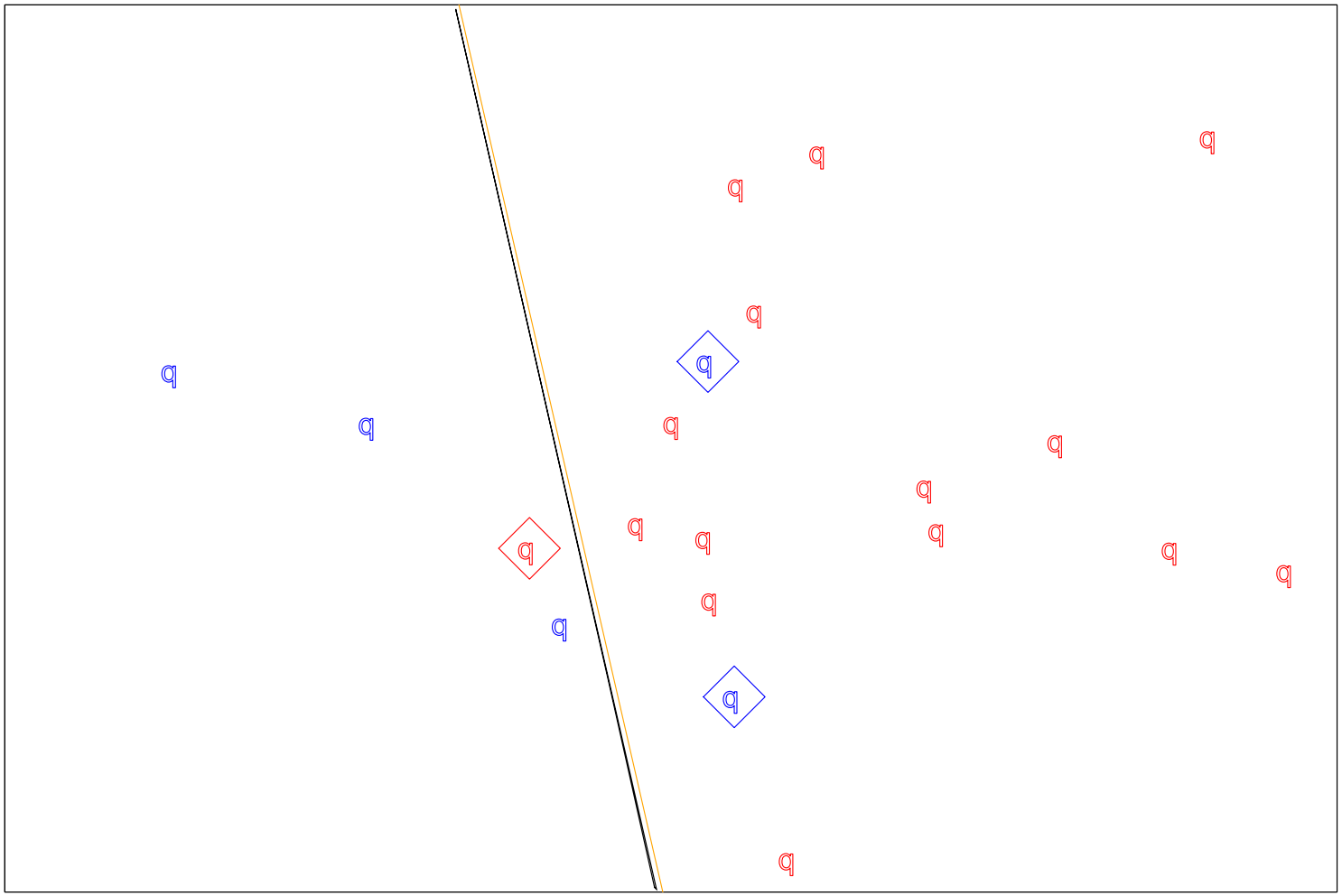
However for applications, we never know the exact probabilities, and instead we must estimate them from the training data

This can produce errors in the estimated set $\hat{Z}$, as illustrated next

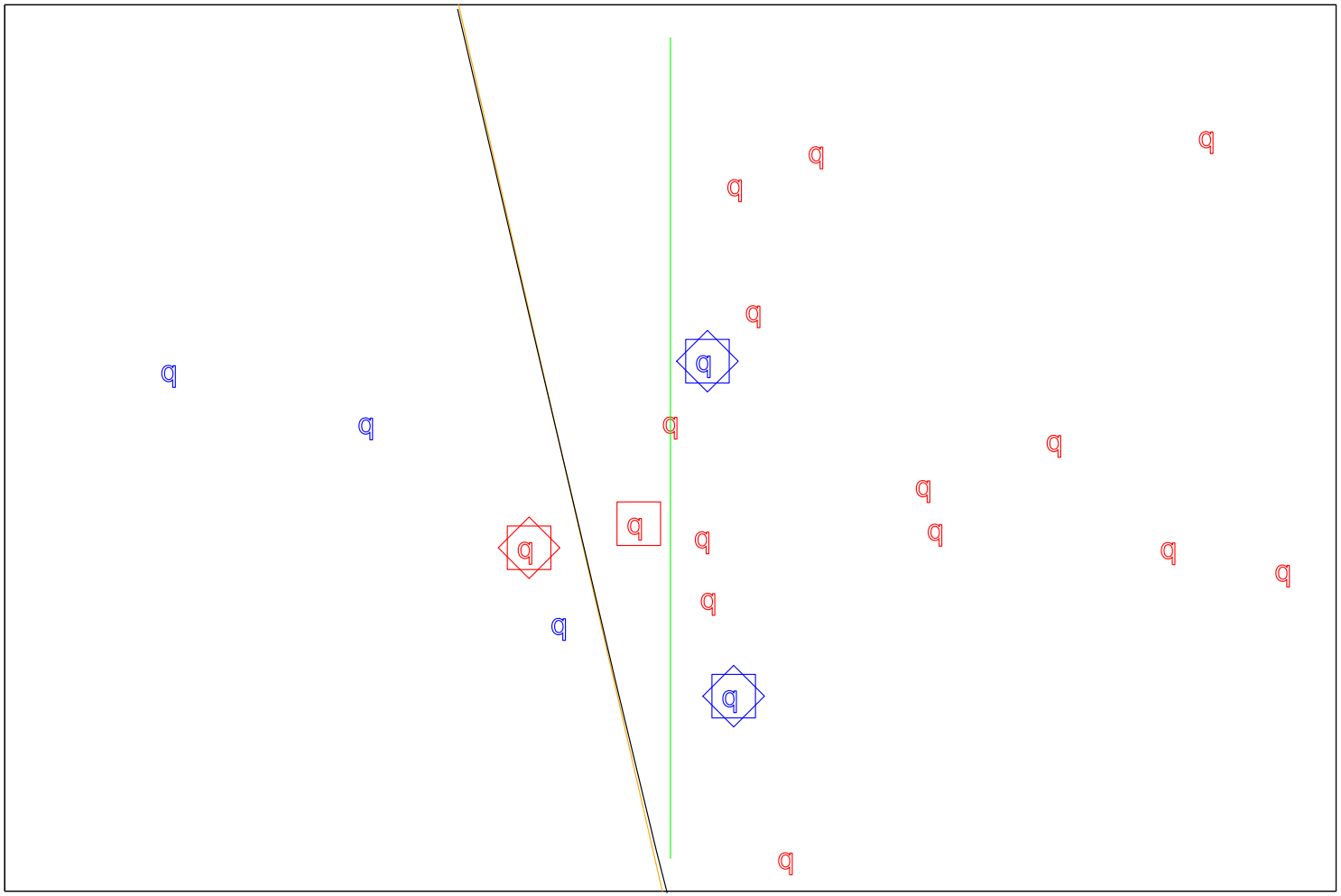
HIL Estimation Algorithms (3)



HIL Estimation Algorithms (3)



HIL Estimation Algorithms (3)



HIL Estimation Algorithms (4)

To estimate class probabilities, we can use any classifier, e.g. something linear like LDA, or something more sophisticated like random forest. This probability-estimating classifier is independent from the main classifier.
When estimating probabilities using the training data, there are various ways to use the data, with different bias-variance tradeoffs:

- In-sample: train and predict on the same data $\mathbf{d}_T$ (danger of overfit)
- Partition, based on K-fold cross-validation [3]. Partition the data $\mathbf{d}_T$ into folds; each fold is a train-test pair; predict on the test part only
- Bootstrap: form many bootstrap samples of the training data $\mathbf{d}_T$. For one sample $\mathbf{d}_{T^b}$, train on $\mathbf{d}_{T^b}$, and predict on $\mathbf{d}_T$. Finally average over the bootstrap samples

HIL Estimation Algorithms (5)

For the second step of HIL actions, we use relabelling, pruning and weighting.

These actions form a spectrum of risk or caution: relabelling is the most aggressive action, whereas weighting is more cautious

For weighting, various different schemes can be used, e.g.

- $H_{w^1}(p_{y_i}) = p_{y_i}$
- $H_{w^2}(p_{y_i}) = 2p_{y_i}$
- $H_{w^e}(p_{y_i}) = \frac{1}{1+exp(-p_{y_i})}$

Experimental Study (1)

We demonstrate the effectiveness of HIL algorithms with a an experimental study

This study focusses on scorecard applications, the prediction of customer default

Two sources of data: UPL data from 1993-1997, and UK credit card data from 2008-2011

Three loss functions: BRAA, AUC, error rate

The primary goal of the study is to establish loss reduction, i.e. to show that

$$L(\theta(\mathbf{d}_H)) < L(\theta(\mathbf{d}_T))$$

Experimental Study (2)

- Scorecards are developed iteratively over time: one year is chosen as training data, the following year as test data
- The scorecard is logistic regression, also used to estimate probabilities. Hence both the original classifier and the HIL classifier are instances of logistic regression
- For each single experiment, comparing classifier losses shows whether a given HIL algorithm performs better, worse or the same than the original classifier
- For a set of n experiments, count the number of wins w. If there is no improvement, this count w has a Binomial distribution

$$w \sim \text{Binomial}(n, p = \frac{1}{2}).$$

This provides a simple statistical test with a p-value to establish significance (large n e.g. 200)

Experimental Study (3)

Method	Win-fraction	P-value	Loss-fraction	Tie-fraction
H_e^b-[ss]-H_{w^2}	0.86818	$<10^{-5}$	0.13182	0
H_e^b-[ss]-H_{w^1}	0.85000	$<10^{-5}$	0.15000	0
H_e^b-[ns]-H_{w^2}	0.89545	$<10^{-5}$	0.10455	0
H_e^b-[ns]-H_{w^1}	0.87272	$<10^{-5}$	0.12728	0

Table: Results for UK credit card data, 2008-2011. The loss function is BRAA. The classifier is logistic regression.

Experimental Study (4)

Method	Win-fraction	P-value	Loss-fraction	Tie-fraction
H_e^b-[ss]-H_{w^2}	0.88182	$<10^{-5}$	0.01818	0
H_e^b-[ss]-H_{w^1}	0.90909	$<10^{-5}$	0.09091	0
H_e^b-[ns]-H_{w^2}	0.88636	$<10^{-5}$	0.11364	0
H_e^b-[ns]-H_{w^1}	0.89545	$<10^{-5}$	0.10455	0

Table: Results for UK credit card data, 2008-2011. The loss function is AUC. The classifier is logistic regression.

Experimental Study (5)

Table: Summary of recommended HIL algorithms for credit risk. The table shows four algorithms, with performance summarised over two datasets and three loss functions.

	UK CC data, 2008-2011			UPL data, 1993-1997		
Algorithm	Braa	AUC	Error	Braa	AUC	Error
H_e^b-[ns]-H_{w^1}	✓	✓	✓	✓	-	✓
H_e^b-[ns-H_{w^2}	✓	✓	✓	✓	-	-
H_e^b-[ss]-H_{w^1}	✓	✓	✓	✓	-	✓
H_e^b-[ss]-H_{w^2}	✓	✓	✓	✓	-	-

Experimental Study (6)

Summarising these results

- HIL weighting is the action which outperforms the original classifier
- The four best-performing HIL algorithms are all closely-related variants. They all use bootstrap resampling to estimate probabilities, then down-weight the improbable labels
- These four algorithms are recommended for credit risk scorecard applications

Conclusion

- HIL appears to provide better scorecards for credit risk
- Initial work suggests a statistical explanation of HIL: the loss reduction comes from reducing the variance of classifier probability estimates
- If you are willing to provide scorecard application data, we're happy to examine the performance boost that HIL can provide

References

Yoav Freund and Robert E. Schapire.

A short introduction to boosting, 1999.

Peter E. Hart.

The condensed nearest neighbor rule.
IEEE Transactions on Information Theory, 14(3):515–516, 1968.

Juan D. Rodriguez, Aritz Perez, and Jose A. Lozano.

A general framework for the statistical analysis of the sources of variance for classification error estimators.
Pattern Recognition, 46(3):855–864, 2013.

"Classifier Loss Reduction by Handling Improbable Labels"; Lewis P G Evans, Niall M Adams and Christoforos Anagnostopoulos, forthcoming

Extra Material (1)

Future work

- Use classifier estimation uncertainty, to hedge the HIL actions. For example use conformal classification and bootstrapping to generate uncertainty estimates for predicted probabilities
- Fine-tune algorithm performance: establish factors for HIL estimation performance (e.g. properties of problem, classifier, and their interaction). Hence model and predict when estimation performance is worse, then take more cautious actions
- Full experimental study to show the HIL achieves loss reduction in just those cases when it also achieves classifier variance reduction
- Experimental investigation of broader scope, with more data and classifiers

Extra Material (2)

Related research

- HIL bears some passing similarity to outlier handling in estimation and regression, but is very different in nature: improbable labels are not noisy, but from the same distribution as the rest of the data
- Data editing (condensing) is somewhat similar to HIL pruning, except that data editing can condense hugely, whereas HIL pruning on average culls only a fraction equal to the Bayes error rate [2]
- Boosting is somewhat related to HIL weighting, in that both approaches weight the training data [1]. However, boosting up-weights examples which are often-misclassified; those same examples would be down-weighted (or equally-weighted) under HIL

Extra Material (3)

Method	Win-fraction	P-value	Loss-fraction	Tie-fraction
H_e^b-[as]-H_{w^2}	0.52333	$<10^{-5}$	0.18167	0.295
H_e^b-[as]-H_{w^1}	0.49333	$<10^{-5}$	0.21	0.29667
H_e^b-[ns]-H_{w^2}	0.48333	$<10^{-5}$	0.23	0.28667
H_e^b-[ss]-H_{w^2}	0.47333	$<10^{-5}$	0.23167	0.295
H_e^b-[ss]-H_{w^1}	0.415	0.00016	0.29333	0.29167
H_e^b-[ns]-H_{w^1}	0.41667	0.00025	0.29833	0.285

Table: Results for UPL data, 1993-1997. The loss function is BRAA. The classifier is logistic regression.

Extra Material (4)

Method	Win-fraction	P-value	Loss-fraction	Tie-fraction
H_e^b-[as]-H_{w^2}	0.75167	$<10^{-5}$	0.24833	0
H_e^b-[as]-H_{w^1}	0.7	$<10^{-5}$	0.3	0

Table: Results for UPL data, 1993-1997. The loss function is AUC. The classifier is logistic regression.

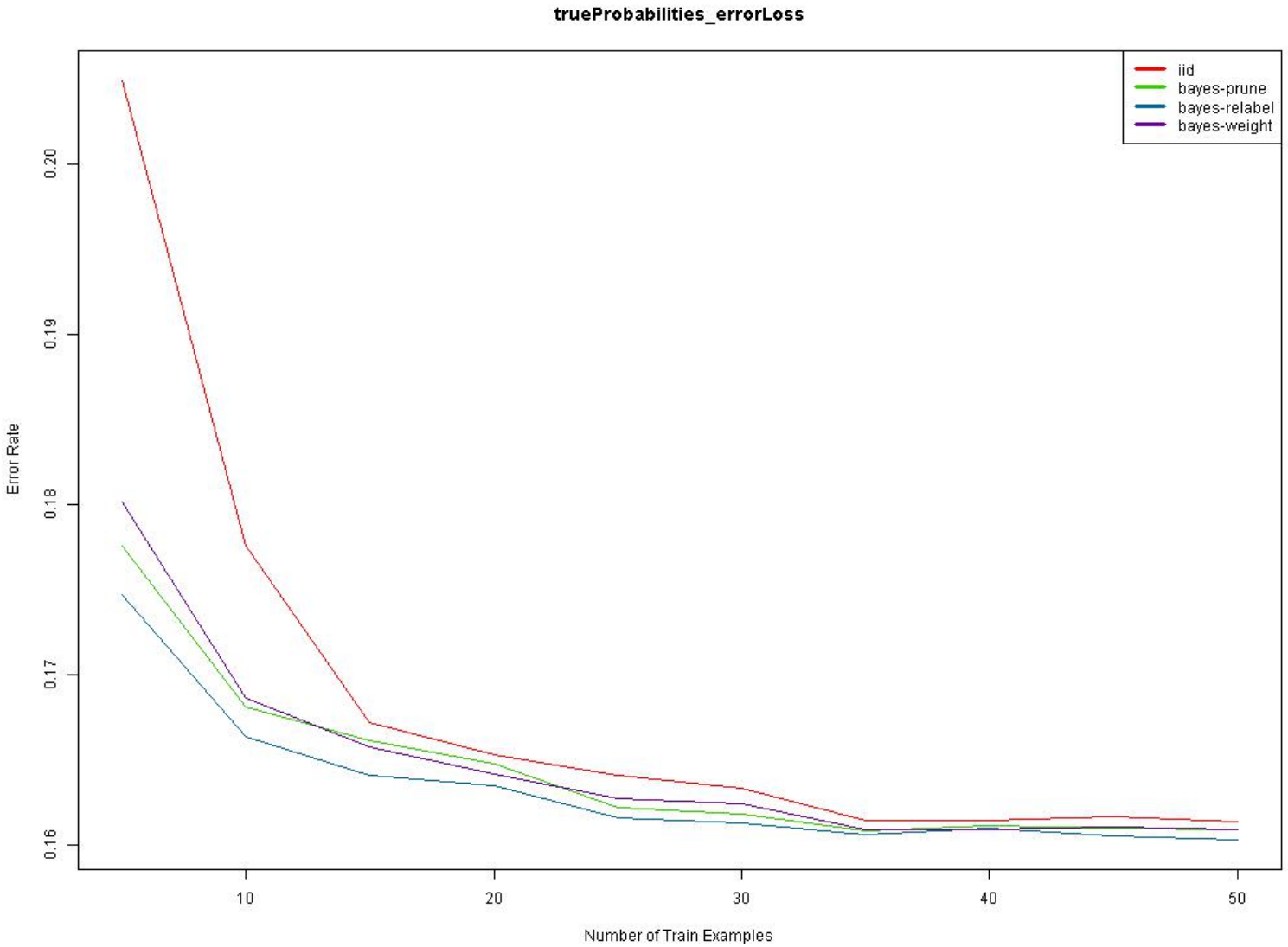
Extra Material (5)

What is the statistical explanation for HIL's loss reduction?
We define classifier variance $v(\theta, n)$ as the variance of its regression function f (for fixed-size training data $\mathbf{d}_T$ of n examples):

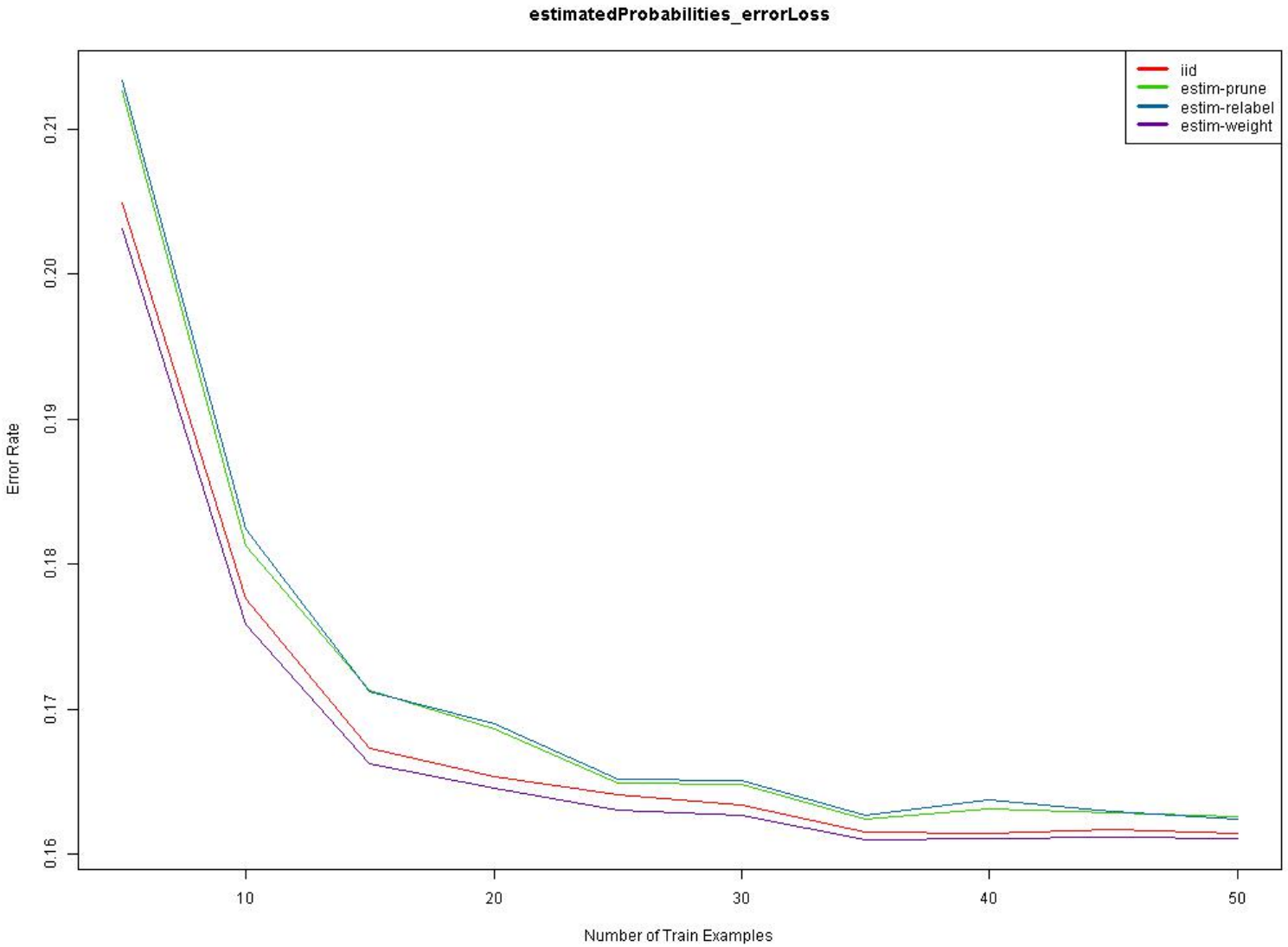
- $$f(\hat{\boldsymbol{\theta}}) = \{\hat{p}_1(\mathbf{x})\}_{\mathbf{x} \in \mathrm{Range}(\mathbf{X})} = \{\hat{p}(c_1|\mathbf{x})\}_{\mathbf{x} \in \mathrm{Range}(\mathbf{X})}.$$
- $$v^x(\theta, \mathbf{x}_i, n) = V\left[\hat{p}_1(\mathbf{x}_i, \hat{\boldsymbol{\theta}}) : \hat{\boldsymbol{\theta}} = \theta(\mathbf{d}_T)\right].$$
- $$v(\theta, n) = \sum_{\mathbf{x} \in \mathrm{Range}(\mathbf{X})} v^x(\theta, \mathbf{x}, n)$$

Initial results suggest that HIL works by reducing the classifier variance $v^r(\theta, n)$: HIL algorithms improve performance in just those cases when they also reduce the classifier variance

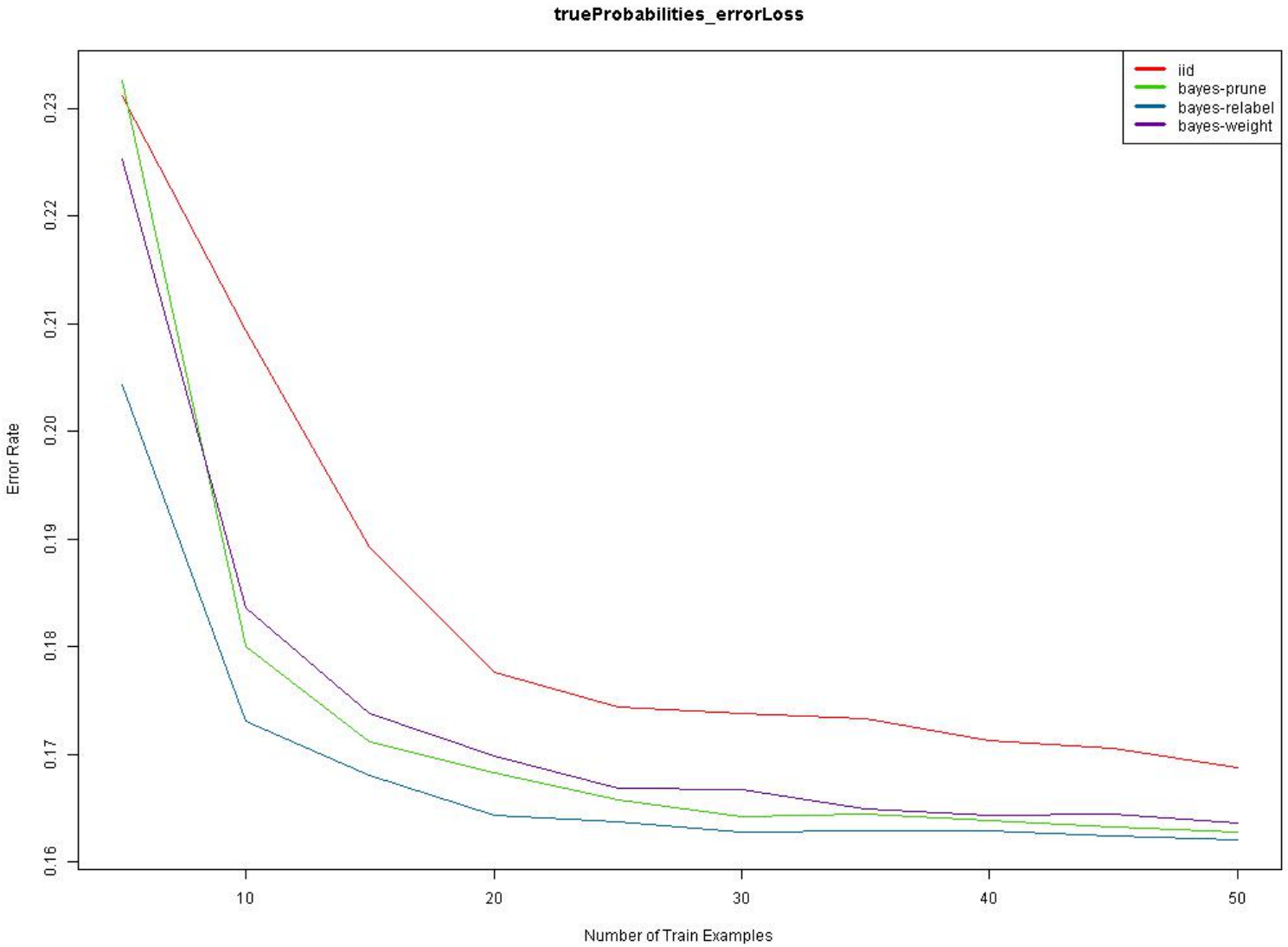
Extra Material (6)



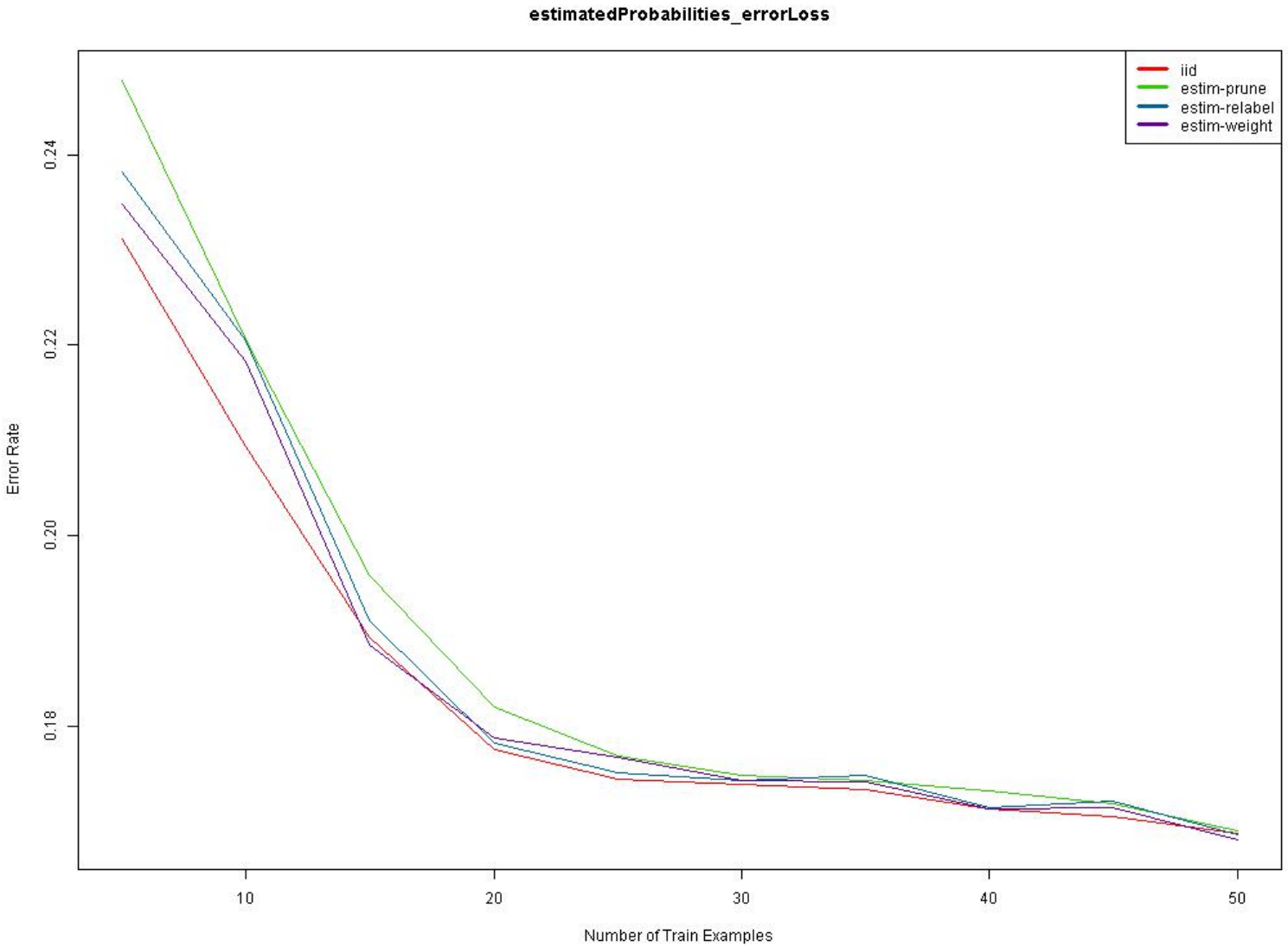
Extra Material (6)



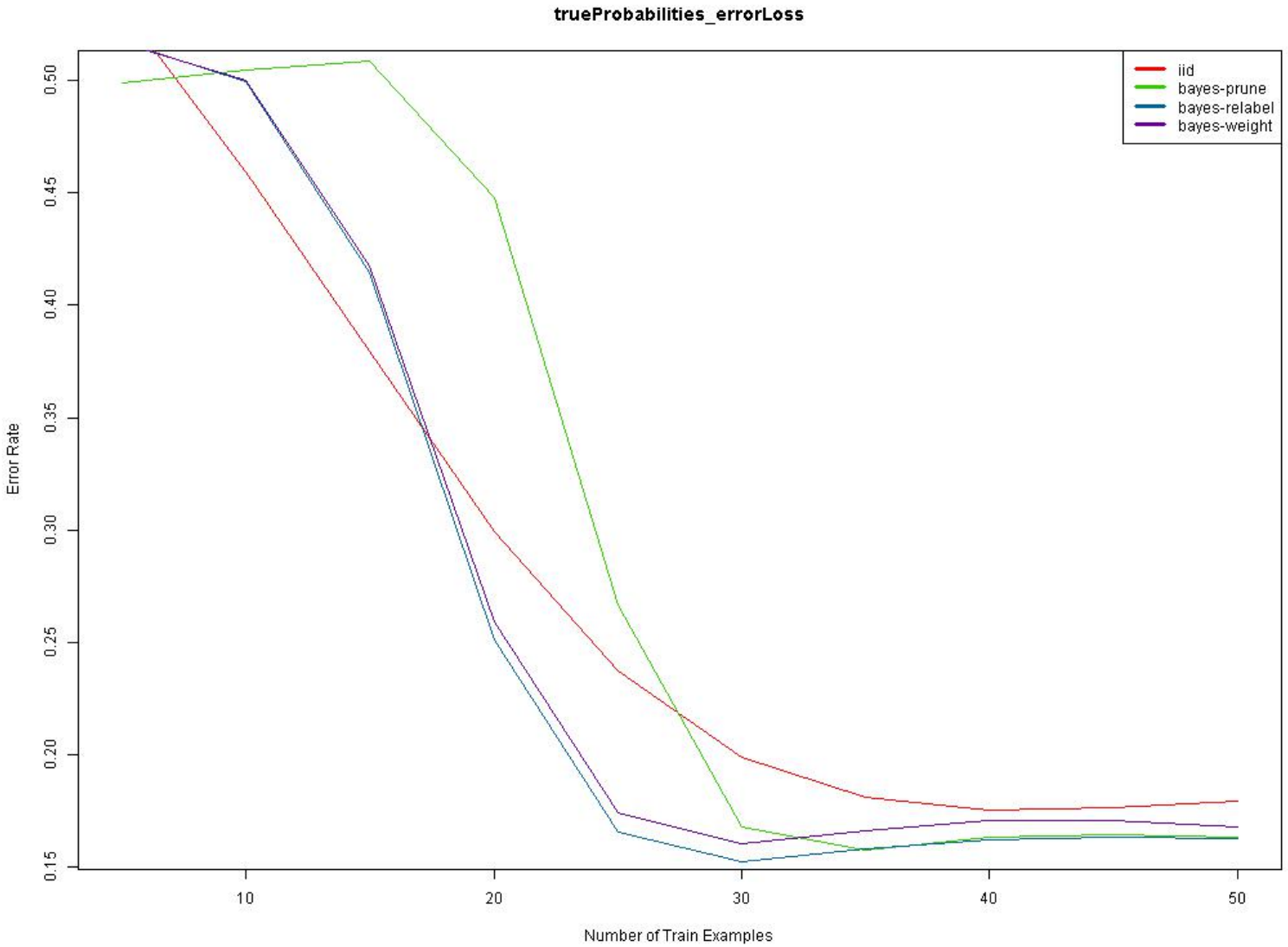
Extra Material (6)



Extra Material (6)



Extra Material (6)



Extra Material (6)