Online Supplement to Iterative Rule Extension for Logic Analysis of Data: a MILP-based heuristic to derive interpretable binary classifiers from large datasets

Marleen Balvert
Department of Econometrics & Operations Research
Zero Hunger Lab, Tilburg School of Economics and Management,
Tilburg University, Tilburg, The Netherlands
m.balvert@tilburguniversity.edu

A Lemmas and proofs

Lemma 1 *Given a set of AND clauses $\mathcal{K}$, then for all $k \in \mathcal{K}$ and $n \in \mathcal{N}$ let $t_{nk} = 1$ if and only if sample n satisfies clause k. Define:*

- $P1 = \{\hat{y} \in [0,1]^{\mathcal{N}} : (1a) and (1b)$ *are satisfied*$\}$;
- $P2 = \{\hat{y} \in [0,1]^{\mathcal{N}} : (2a) and (2b)$ *are satisfied*$\}$.

Then $P2 \subseteq P1$.

Proof. First note that equations (1a) and (2a) are identical. Second, (2b) $\Rightarrow$ (1b) is trivial. In order to see that (1b) does not imply (2b), consider the following example. Suppose that $\mathcal{K} = \{1, 2\}$, $t_{n1} = 1$ and $t_{n2} = 0$ for some $n \in \mathcal{N}_0$. Then constraint (2b) enforces $\hat{y}_n = 1$, while constraint (1b) only requires $\hat{y}_n \geq 0.5$.

Lemma 2 *Define:*

- $P3 = \{t \in [0,1]^{N \times K},\ s \in [0,1]^{J \times K} :$ *(3a) and (3b) are satisfied* $\forall j \in \mathcal{J}, \forall n \in \mathcal{N}, \forall k \in \mathcal{K}\}$;
- $P4 = \{t \in [0,1]^{N \times K},\ s \in [0,1]^{J \times K} :$ *(4a) and (4b) are satisfied* $\forall j \in \mathcal{J}, \forall n \in \mathcal{N}, \forall k \in \mathcal{K}\}$.

Then $P4 \subseteq P3$.

Proof. Since (3b) and (4b) are identical, it suffices to show that (4a) $\Rightarrow$ (3a), but (3a) $\Rightarrow$ (4a) does not hold.
(4a) $\Rightarrow$ (3a) Recall (4a): $t_{nk} - (X_{nj} - 1)s_{kj} \leq 1 \quad \forall j \in \mathcal{J}$ Summing both sides over $j \in \mathcal{J}$ directly gives (3a).
(3a) does not imply (4a) Consider the following example. Let $\mathcal{J} = \{1, 2\}$, $s_{k1} = \frac{1}{4}$, $s_{k2} = \frac{3}{4}$ for some $k \in \{1, ..., K\}$, $X_{n1} = 0$ and $X_{n2} = 0$ for some $n \in \mathcal{N}$. Then (3a) implies: $2 \cdot t_{nk} + \frac{1}{4} + \frac{3}{4} \leq 0 \Leftrightarrow t_{nk} \leq \frac{1}{2}$, while this is not feasible with respect to constraint (4a): $\begin{cases} t_{nk} + \frac{1}{4} \leq 1 \\ t_{nk} + \frac{3}{4} \leq 1 \end{cases} \Leftrightarrow t_{nk} \leq \frac{1}{4}$.

B Relaxing integrality of variables

Depending on the model formulation, some of the integrality constraints can be relaxed. Table 1 in the original paper provides an overview of the continuous and the binary variables in each of the six model formulations.

In order to understand for which variables the integrality constraint can be relaxed, each of the constraints will be discussed separately. For constraints (1), provided that t_{nk} are binary, the integrality on $\hat{y}_n$ can be relaxed for $n \in \mathcal{N}_1$ but not for $n \in \mathcal{N}_0$. Let us first consider constraint (1a), i.e. $\hat{y}_n - \sum_{k \in K} t_{nk} \leq 0 \ \forall n \in \mathcal{N}_1$. As $\hat{y}_n$ is maximized for $n \in \mathcal{N}_1$, we have $\hat{y}_n = \sum_{k \in K} t_{nk}$ when $\sum_{k \in K} t_{nk} \leq 1$, and $\hat{y}_n = 1$ otherwise. Since t_{nk} is integer, $\hat{y}_n$ will be integer as well, hence the integrality constraint on $\hat{y}_n$ for $n \in \mathcal{N}_1$ can be relaxed. For $n \in \mathcal{N}_0$ we need to consider constraint (1b): $K\hat{y}_n - \sum_{k \in K} t_{nk} \geq 0$. Since $\hat{y}_n$, $n \in \mathcal{N}_0$ is minimized, relaxing the integrality constraint implies $\hat{y}_n = \frac{1}{K} \sum_{k \in K} t_{nk}$ as $\sum_{k \in K} t_{nk}$ is nonnegative. Hence, the constraint $\hat{y}_n \in \{0, 1\}$ cannot be relaxed for $n \in \mathcal{N}_0$.

Constraints (2a) are identical to constraints (1a), hence also for constraints (2) one can relax the integrality on $\hat{y}_n$ for $n \in \mathcal{N}_1$, provided that t_{nk} are binary. Also the integrality on $\hat{y}_n$ for $n \in \mathcal{N}_0$ can be relaxed when using constraints (2): $\hat{y}_n - t_{nk} \geq 0 \; \forall k \in \mathcal{K}, \; \forall n \in \mathcal{N}_0$ implies that $\hat{y}_n$ is equal to the minimum t_{nk} over all k, which is binary.

When looking at constraints (3), first note that X_{nj} is binary by definition. For constraints (3a), i.e. $J \cdot t_{nk} + \sum_{j \in \mathcal{J}} (1 - X_{nj}) s_{kj} \leq J \; \forall k \in \mathcal{K}, \; \forall n \in \mathcal{N}_1$, consider the case where $J - \sum_{j \in \mathcal{J}} (1 - X_{nj}) s_{kj} < J$. Then t_{nk} should be equal to zero, though when relaxing the integrality on t_{nk} it can become fractional. Hence, the restriction $t_{nk} \in \{0, 1\}$ for $n \in \mathcal{N}_1$ cannot be relaxed. This restriction can however be relaxed for $n \in \mathcal{N}_0$, provided that s_{kj} is integer. Consider constraints (3b): $t_{nk} + \sum_{j \in \mathcal{J}} (1 - X_{nj}) s_{kj} \geq 1 \; \forall k \in \mathcal{K}, \; \forall n \in \mathcal{N}_0$. Since t_{nk} is (indirectly) minimized for $n \in \mathcal{N}_0$ constraints (3b) and $t_{nk} \in [0, 1]$ together imply $t_{nk} = 1 - \sum_{j \in \mathcal{J}} (1 - X_{nj}) s_{kj}$ when $\sum_{j \in \mathcal{J}} (1 - X_{nj}) s_{kj} \leq 1$, and $t_{nk} = 0$ otherwise. Since $\sum_{j \in \mathcal{J}} (1 - X_{nj}) s_{kj}$ is integer, the integrality constraint on t_{nk}, $n \in \mathcal{N}_0$ can be relaxed.

When using constraints (4) the integrality on t_{nk} can be relaxed for all $n \in \mathcal{N}$ provided that s_{kj} are binary. First note that t_{nk} is (indirectly) maximized for $n \in \mathcal{N}_1$. Constraints (4a) then imply that $t_{nk} = \min_{j \in \mathcal{J}} (1 - X_{nj}) s_{kj}$ for $n \in \mathcal{N}_1$, which is binary. Constraints (4b) are identical to constraints (3b), which imply that $t_{nk}, n \in \mathcal{N}_0$ are binary, see above.

C Effect of data sizes on runtime of $(BP1)$

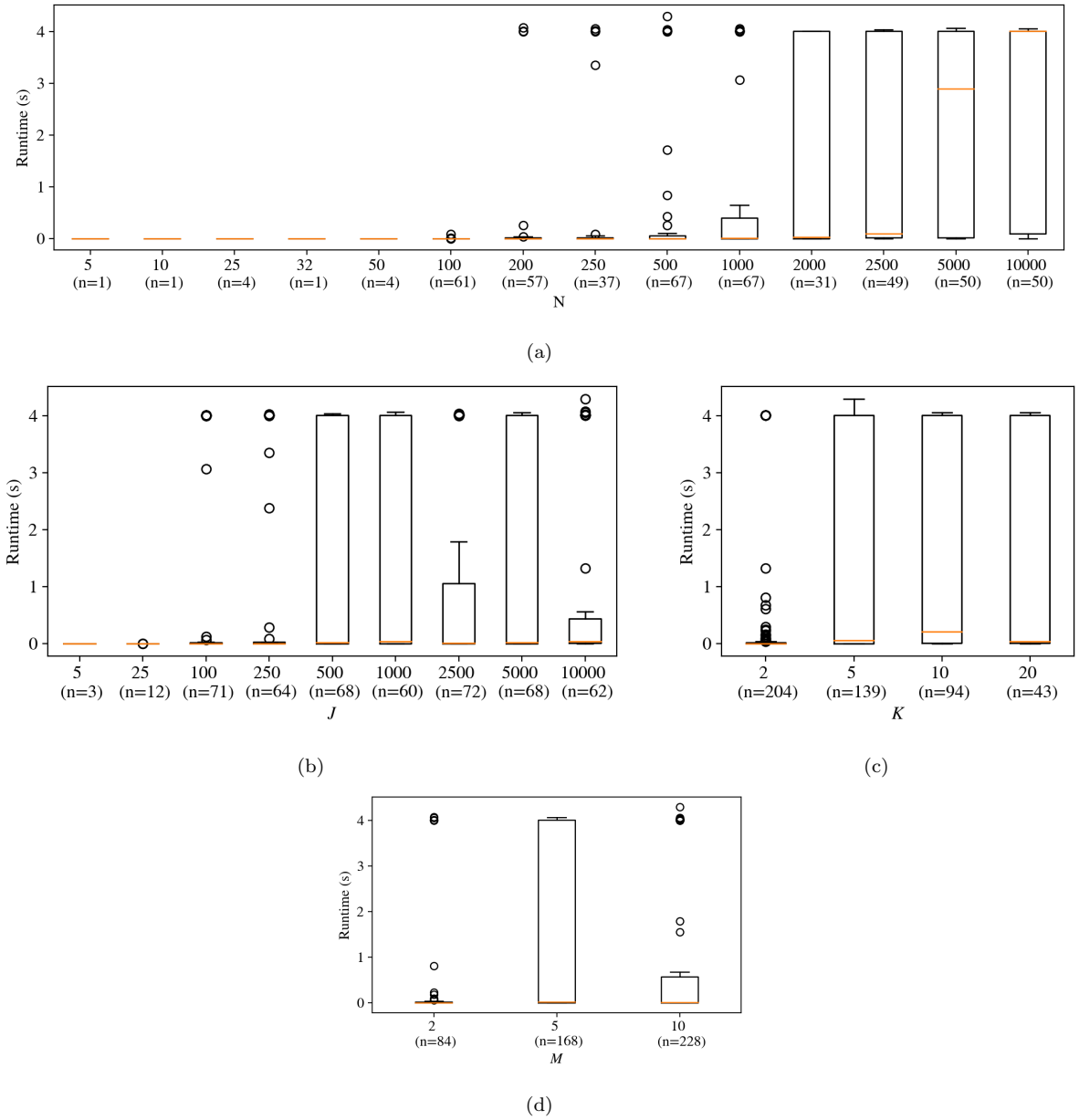


Figure 1: Boxplots of runtimes obtained with $(BP1)$ for various no noise datasets, aggregated by (a) the number of samples N, (b) the number of features J, (c) the number of AND clauses K and (d) the maximum number of features per clause M. The orange bar denotes the average value and n denotes the number of instances for each parameter value. The figures show that N has a major impact on the performance of $(BP1)$. J and K influence its performance as well as small values for J (< 500) and K ($= 2$) yield low runtimes, while M does not seem to be related to model performance.

Table 1: Parameter values used for creating synthetic datasets. Each combination of the indicated parameter values is used to generate one dataset. N denotes the number of samples, J the number of sample features, K the number of AND clauses, M the maximum number of features in an AND clause and ε the error rate. Note that ε is not used for the datasets without noise.

Parameter	No noise collection	Noise collection
N	100, 500, 1000, 2500, 5000, 10000	500, 1000, 5000, 10000
J	100, 500, 1000, 2500, 5000, 10000	500, 1000, 5000, 10000
K	2, 5, 10	2, 5, 10
M	10	10
ε	n.a.	0.01, 0.025, 0.05

D Generating datasets

D.1 Generating synthetic data

For each dataset, the input matrix X and the Boolean phrase in DNF are generated randomly for a given choice of N, J, K and M. The input matrix X is an $N \times J$ matrix where each element is randomly drawn from a Bernoulli distribution with $p = 0.5$. No duplicate rows are allowed, that is, no two samples are allowed to be the same. In order to generate a Boolean phrase in DNF, K random AND clauses are generated. For each AND clause first a random number $m \in \{1, ..., M\}$ is drawn from a uniform distribution. This is the number of sample features to be included in the AND clause. Then m sample features are randomly drawn without replacement from the set of features, again using a uniform distribution. This gives a Boolean phrase in DNF that consists of K AND clauses that each contain at most M sample features. The output vector y is then obtained by applying the Boolean phrase to the input matrix X. If the error rate ε for this dataset is non-zero, y_n is inverted for $\varepsilon \cdot N$ randomly selected samples. For each dataset, test data was generated by creating an additional matrix $X_t \in [0, 1]^{N \times J}$ that contains no rows identical to those in X. Class labels corresponding to X_t were generated using the same Boolean phrases in DNF as for the original (training) dataset, and random errors were applied.

D.2 Genotype data from the 1000 Genomes Consortium

The genomic datasets all contain the same genetic information of 2504 individuals from the 1000 Genomes Project (1000 Genomes Project Consortium 2015), obtained from Bayat et al. (2020). As the full data contains approximately 80 million sample features, Bayat et al. (2020) provide three subsets, the smallest of which contains approximately 500,000 sample features[1]. This data was further reduced in three steps. First LD pruning was applied using PLINK (Purcell 2007, Purcell et al. 2007) with a window size of 100kb, a step size of 50 and a VIF threshold of 0.5. Genetic features that are physically close to each other on the genome are often highly correlated due to inheritance principles, which is referred to as linkeage disequilibrium (LD). Based on this structure sample features that are in high LD with each other, i.e., that are strongly correlated with each other, are filtered such that no such high correlations remain in the data. PLINK is an often used software tool for LD pruning. After the first step 477,803 sample features remain. Second, only those sample features with a minor allele frequency (MAF, the frequency of the least frequent variant) between 10% and 50% are kept, leaving 42,451 sample features. In the third step only chromosomes 1 and 3 were selected to obtain the final genomic data of 9,633 features. All features are tertiary, i.e. they are 0, 1 or 2, indicating how often the minor allele occurs in an individual. The features were binarized by converting every occurrence of a 2 into a 1. As a result the new features reflect whether the minor allele occurs at least once in an individual. This dataset is split into a training set that contains 2002 samples and a test set containing 502 samples.

D.3 Class labels for the 1000 Genomes dataset

Two collections of datasets are generated with the 1000 Genomes input data. As no true class labels are available we have to use synthetic labels. For the first collection of datasets, referred to as the DNF collection, class labels are generated in the same way as for the two collections of synthetic data, hence the relationship between input data and class labels is a Boolean phrase in DNF. For the second collection, referred to as

[1]This data is obtained in the file "subset.3.vcf.bgz" provided on https://ftp.cngb.org/pub/gigadb/pub/10.5524/100001_01000/100759/$Dataset Subsets, accessed on April 20, 2022$.

PEPS, we use the synthetic class labels generated by Bayat et al. (2020), see their work for further details. Note that these labels are generated with a different form than Boolean phrases in DNF.

E Hyperparameter tuning

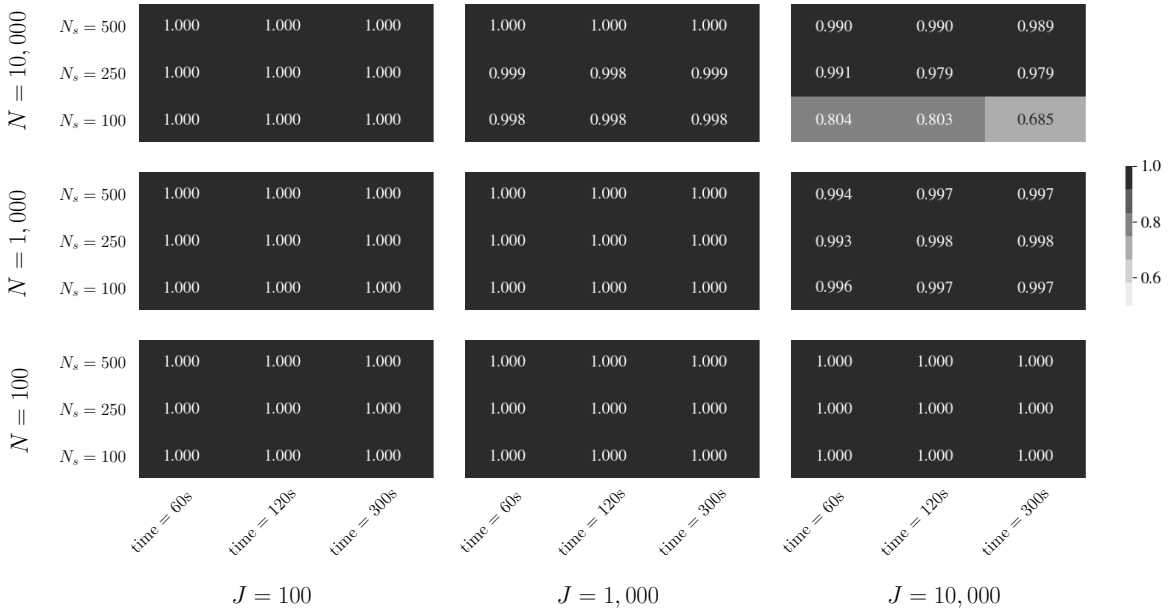


Figure 2: Normalized objective values averaged over multiple noiseless datasets, split by N and J. Various values for N_s and the time limit of the sub problem are tested: $N_s \in \{100, 250, 500\}$ and time limit $\in \{60s, 120s, 300s\}$.

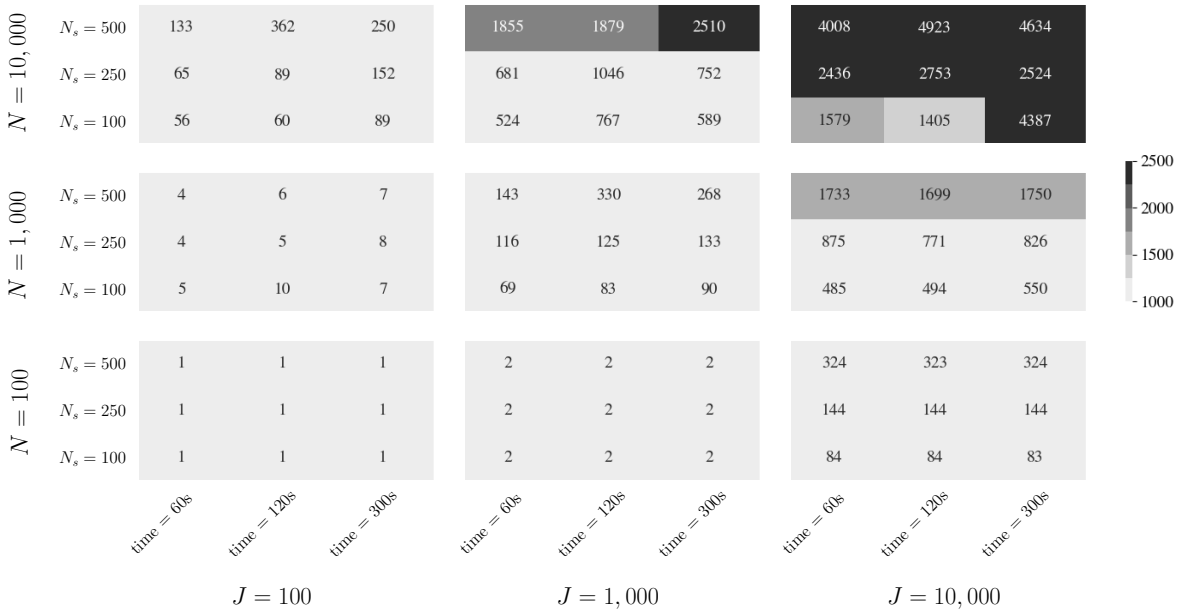


Figure 3: Runtimes averaged over multiple noiseless datasets, split by N and J. Various values for N_s and the time limit of the sub problem are tested: $N_s \in \{100, 250, 500\}$ and time limit $\in \{60s, 120s, 300s\}$.

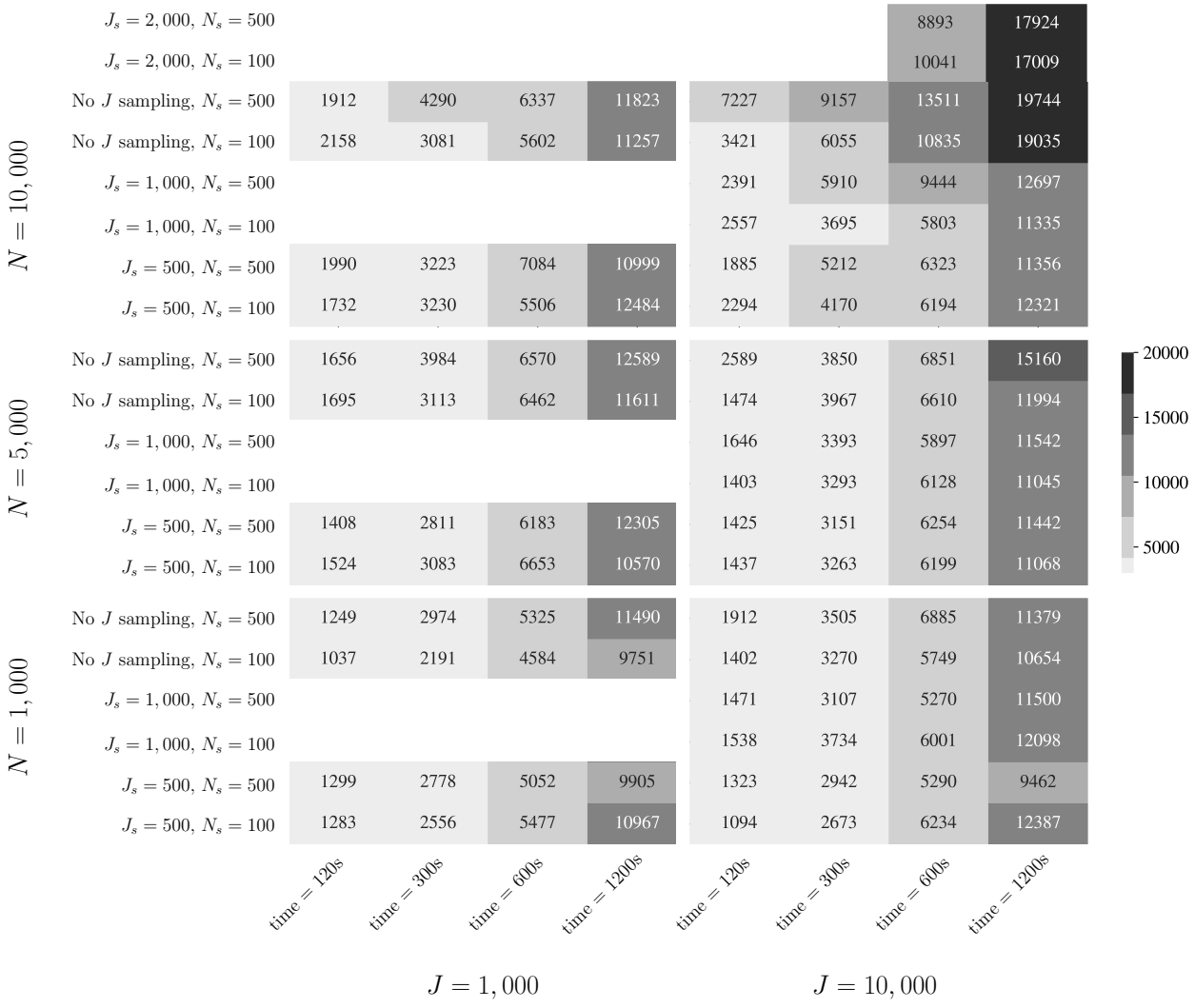
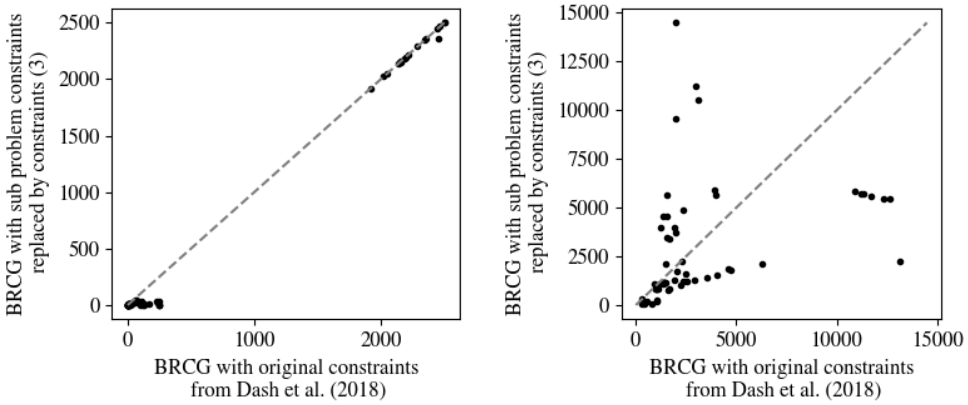


Figure 4: Runtimes averaged over multiple noisy datasets, split by N and J. Various values for N_s and the time limit of the sub problem are tested: $N_s \in \{100, 500\}$, $J_s \in \{500, 1000, J\}$ - where $J_s = J$ refers to no feature subsampling - and time limit $\in \{120s, 300s, 600s, 1200s\}$. Note that for datasets where $J = 1,000$ subsampling 1,000 features is identical to no feature subsampling, hence these runs are omitted.

F Comparison of sub problem constraints in BRCG

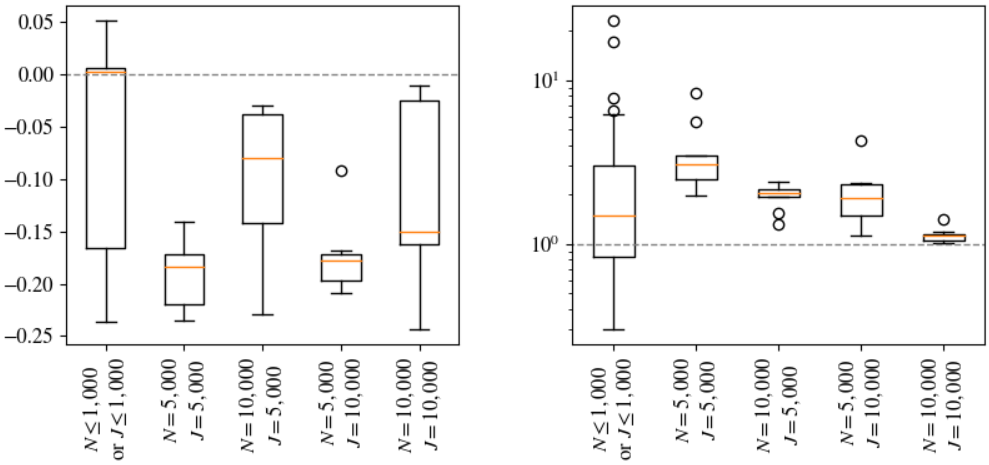


(a) Hamming loss

(b) Runtime (s)

Figure 5: Comparison of the performance of BRCG where the sub problem has its original constraints versus BRCG with the constraints in the sub problem replaced by constraints eq:AND1 in terms of (a) Hamming loss and (b) runtime in seconds.

G Performance comparison IRELAND and BRCG per instance



(a) Normalized objective value obtained with IRELAND minus normalized objective value obtained with BRCG.

(b) Runtime (s) for IRELAND divided by runtime (s) for BRCG.

Figure 6: Boxplots of (a) difference in normalized objective value between IRELAND and BRCG and (b) ratio of runtime (s) for IRELAND and BRCG per instance. Negative values imply that IRELAND outperform BRCG, positive values imply that BRCG outperforms IRELAND.

H Results on test data

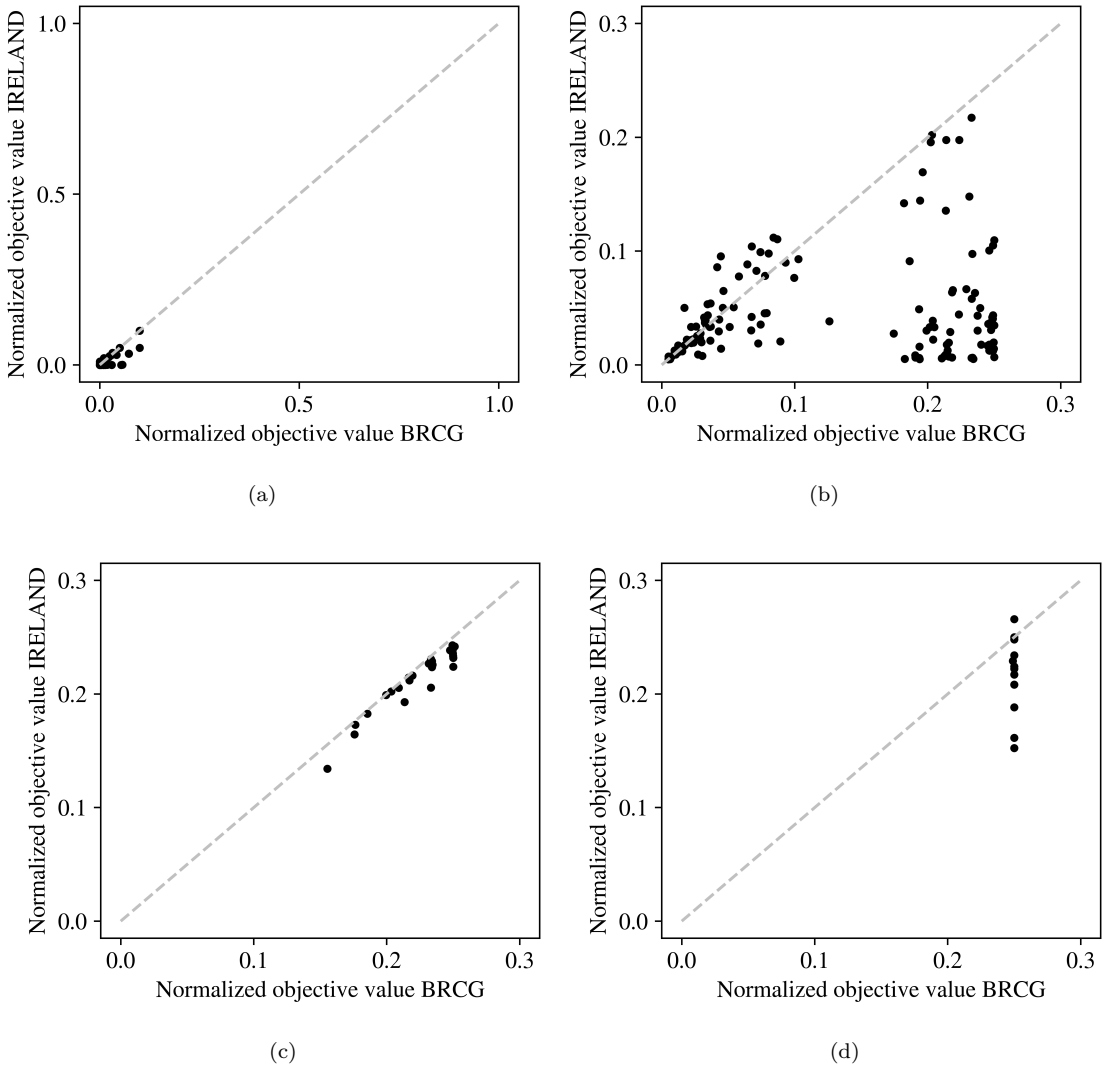


Figure 7: Comparison of the performance of BRCG versus IRELAND on test datasets in terms of normalized objective function value for synthetic datasets with noise (a), the 1000 Genomes data with Boolean rule class labels (b) and the 1000 Genomes data with class labels obtained from Bayat et al. (2020) (c). Each dot represents a dataset for which the normalized objective value of BRCG is shown on the horizontal axis, and the normalized objective value of IRELAND is shown on the vertical axis. The dashed line indicates equal performance between the methods.

I Illustration of Boolean rules in DNF identified by IRELAND

Clause	Features	In true Boolean rule	Found by IRELAND	Found by BRCG	tp	fp
1	50, 66	yes	yes	yes	230	5
2	159, 264, 278	yes	yes	yes	123	2
3	29,81,255,367,392	yes	yes	yes	23	1
4	46,63,210,369,469	yes	yes	yes	32	0
5	158,252,273,335,369,393,424	yes	no	no	3	0
6	67,191,227,309,369,385,464	no	yes	no	11	4
7	10,51,126,251,324,345,431	no	no	yes	12	1

Table 2: AND clauses in the true Boolean rule underlying Inst362, and AND clauses found by IRELAND for this instance. This instance contains 1,000 samples and 500 features. The true underlying Boolean rule consists of five AND clauses, and the error on y is 0.025.

Clause	Features	In true Boolean rule	Found by IRELAND	Found by BRCG	tp	fp
1	3069	yes	yes	yes	498	4
2	1072,4482	yes	yes	yes	236	4
3	1366,1572,1752	yes	yes	yes	112	0
4	819,1089,1826,2516,2650, 3517,4160,4218,4668	yes	no	no	2	0
5	13,1335,1352,2533,3880	yes	no	no	26	1
6	408,1146,1348,2223,3345, 3616,4632	no	yes	no	3	0
7	1095,1352,1482,2630,3222, 3473,4014,4092,4766,4826	no	yes	no	10	1
8	246, 709, 1627, 2630, 4092, 4376, 4826	no	no	yes	16	0
9	515, 1323, 2150, 2693, 3142, 3944	no	no	yes	23	0

Table 3: AND clauses in the true Boolean rule underlying Inst1003 and AND clauses found by IRELAND for this instance. This instance contains 1,000 samples and 5,000 features. The true underlying Boolean rule consists of five AND clauses, and the error on y is 0.01.

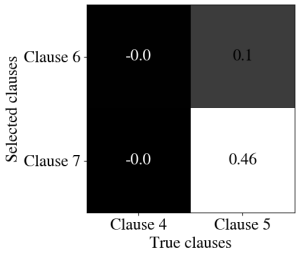


Figure 8: Correlation between clauses 4 and 5 (in the true underlying Boolean rule but not selected by IRELAND) and clauses 6 and 7 (selected by IRELAND but not in the true underlying Boolean rule) for Inst1003.

Clause	Features	In true Boolean rule	Found by IRELAND	Found by BRCG	tp	fp
1	2960,7229	yes	yes	no	276	7
2	845,2122,2684,2909,9554	yes	no	no	11	0
3	2878,3560,3954,5349,6063,7684,8768	yes	no	no	4	0
4	2604,5641,6388,7121,7642,9306	yes	no	no	3	1
5	1107,1193,3981,6444,6707,7017,7512	yes	no	no	2	0
6	2587,4475,6810,9006	yes	no	no	1	0

Table 4: AND clauses in the true Boolean rule underlying an instance from the genotype dataset with phenotypes generated by a Boolean rule in DNF, and AND clauses found by IRELAND for this instance. For the genotype instances, BRCG did not select any clauses.

J Model complexities

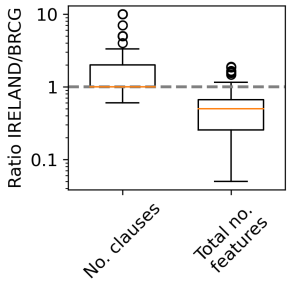


Figure 9: Comparison of the complexities of the rules obtained with IRELAND and BRCG. The figure shows boxplots of the ratio of the number of clauses in a rule generated by IRELAND versus the number of clauses in a rule generated by BRCG (left), and the ratio of the number of features included in a rule generated by IRELAND versus the number of features included in a rule generated by BRCG.

References

1000 Genomes Project Consortium (2015) A global reference for human genetic variation. *Nature* 526.

Bayat A, Piotr S, O'Brian AR, Dunne R, Hosking B, Jain Y, Hosking C, Luo OJ, Twine N, Bauer DC (2020) Variantspark: Cloud-based machine learning for association study of complex phenotype and large-scale genomic data. *GigaScience Database* 9(8).

Purcell S (2007) PLINK version 1.90b6.21.

Purcell S, Neale B, Todd-Brown K, Thomas L, Ferreira MA, Bender D, Maller J, Sklar P, De Bakker PI, Daly MJ, Sham PC (2007) Plink: a tool set for whole-genome association and population-based linkage analyses. *The American journal of Human Genetics* 81:559–575.