Séminaire de Recrutement au Poste de Professeur en Machine Learning / Statistique CREST, ENSAI

Mokhtar Z. Alaya
Équipe APP - LITIS, Université Rouen Normandie
https://mzalaya.github.io/

29 Mai 2020

Education / Professional Experience

2011 — 2012 — 2016 — 2019 → now

- **2010–2011: MSc Probability and Random Models,** *Université Pierre et Marie Curie*
  Research Internship, *INRIA-ENS and Orange Labs*

- **2011–2012: MSc Statistics**, *Université Pierre et Marie Curie*
  Research Internship, *LSTA, Université Pierre et Marie Curie*

- **2012–2016: PhD candidate**, *LSTA, Université Pierre et Marie Curie*
  Title: *Segmentation of Counting Processes and Dynamical Models*
  Supervision: Stéphane Gaiffas and Agathe Guilloux

- **2015–2016: ATER**, *LSTA, Université Pierre et Marie Curie*

- **2016–2017: ATER**, *Modal'X, Université Paris Nanterre*

- **2017–2018: Postdoc 1**, *Modal'X, Université Paris Nanterre*
  Laureat Postdoc Fellowship Fondation Sciences Mathématiques de Paris

- **2019–2021: Postdoc 2**, *APP-TEAM, LITIS, Université Rouen Normandie*
  ANR-Project OATMIL (OptimAl Transport for MachIne Learning)

Statistics / ML

High-Dimensional Statistics

Learning the intensity of Time Events with Change Points
IEEE. Trans. Inform. Theory'15

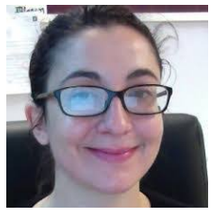
Binarsity: A Penalization for One-Hot Encoded Features in Linear Supervised Learning
JMRL'19

Collective Matrix Completion
JMLR'19

Binacox: Automatic Cut-Points Detection in High-Dimensions Cox Model
submitted to Biometrika'20

High-Dimensional Time-Varying Aalen and Cox Models
in revision in Nonparam. Stat.'20


Pr. S. Gaiffas


Pr. A. Guilloux


Dr. O. Klopp


Dr. S. Bussy


Dr. T. Allart


Dr. S. Lemler

Optimal Transport for Machine Learning


Pr. A. Rakotomamonjy


Pr. G. Gasso


Dr. M. Bérar


Dr. L. Chapel

Screening Sinkhorn Algorithm for Regularized Optimal Transport
NeurIPs'19

Non-aligned Distribution using Metric Measure Embedding and Optimal Transport
submitted to NeurIPS'20

Partial Gromov-Wasserstein with Applications on Positive-Unlabeled Learning
submitted to ICML'20

Open Set Domain Adaptation using Optimal Transport
submitted to ECML-PKDD'20

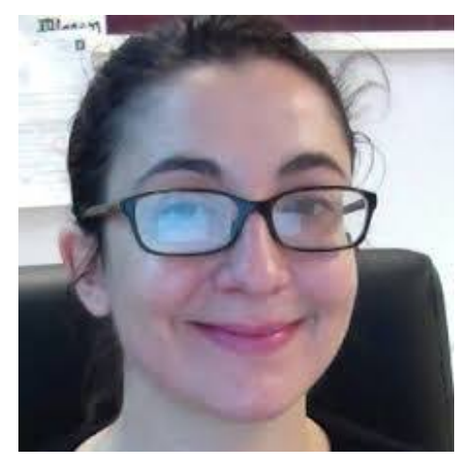
Scientific Production

Binarsity
A Penalization for One-Hot Encoded Features in Linear Supervised Learning
(JMLR'19)

Joint work with …

Dr. Simon Bussy
Researcher at INSERM
Co-funder of Califrais

Pr. Stéphane Gaiffas
Professor at LPSM
Univ. Paris Diderot



Pr. Agathe Guilloux
Professor at LaMME
Univ. Evry Val d'Essonne

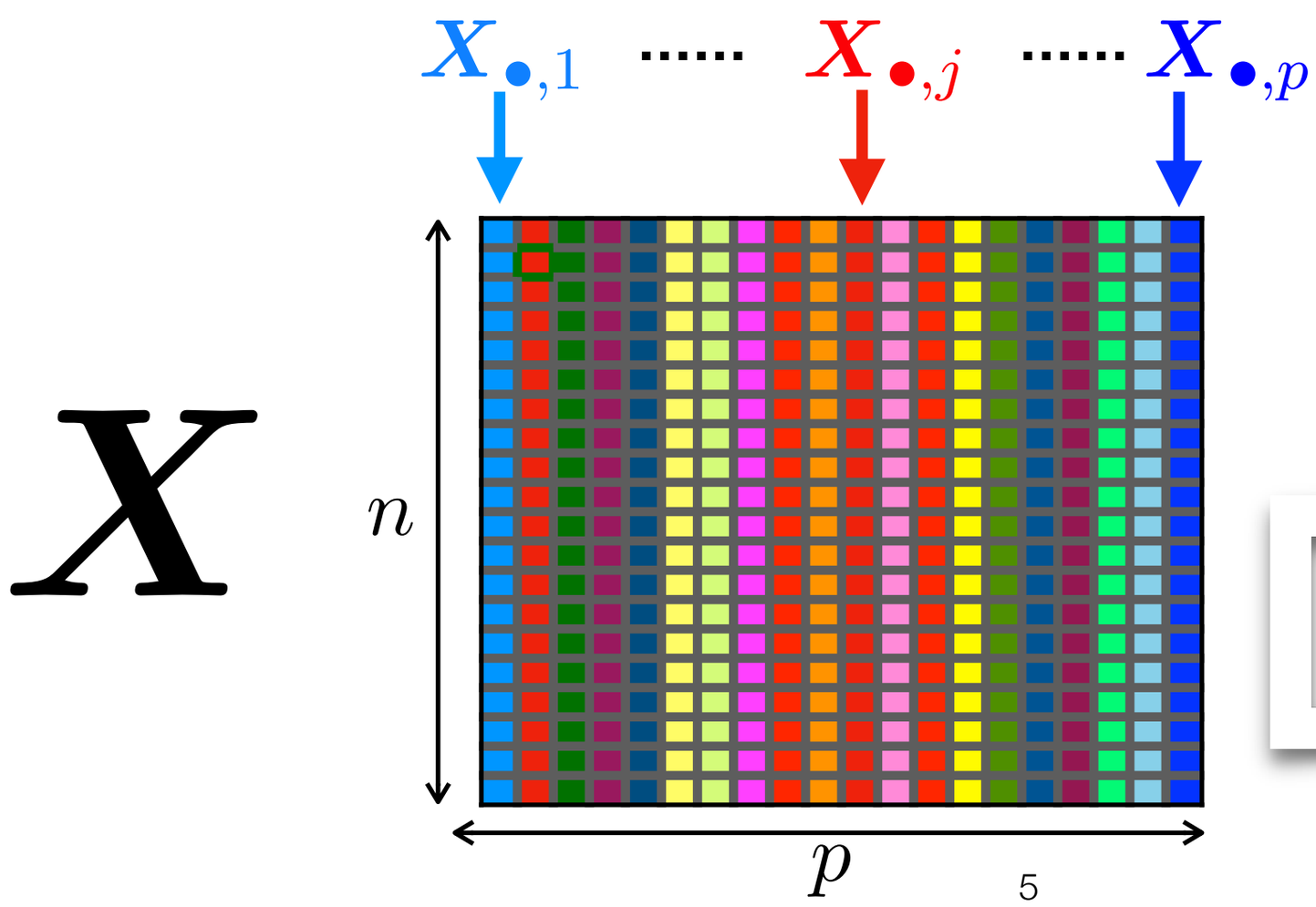
Supervised Learning: Setting

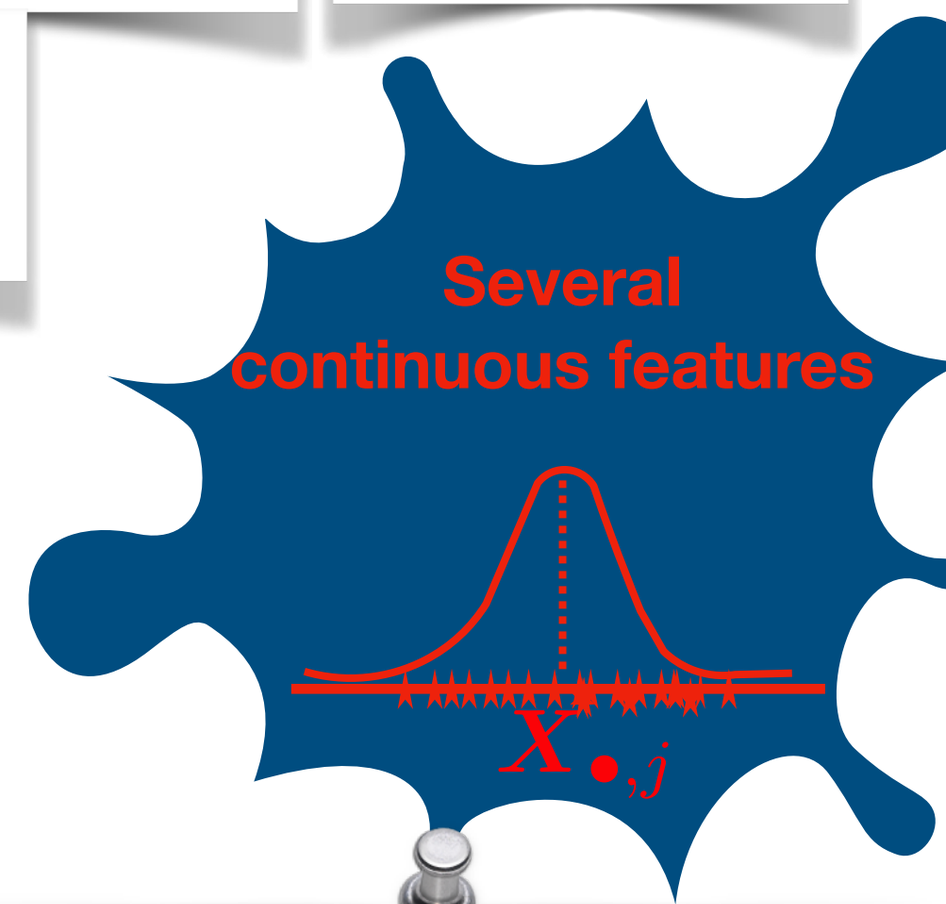
Supervised training dataset
$\mathcal{D}_n = \{(x_i, y_i) : i = 1, \ldots, n\}$

With features
$x_i = [x_{i,1}, \ldots, x_{i,p}]^\top \in \mathbb{R}^p$

And labels
$y_i \in \mathcal{Y} \subset \mathbb{R}$

Features matrix
$$\boldsymbol{X} = [x_{i,j}]_{1 \leq i \leq n; 1 \leq j \leq p} \in \mathbb{R}^{n \times p}$$





A well know-trick:
One-Hot Encoding

(Lieu et al., 2002);

Wu and Coggeshall, 2012)

One-Hot Encoding: Features Binarization

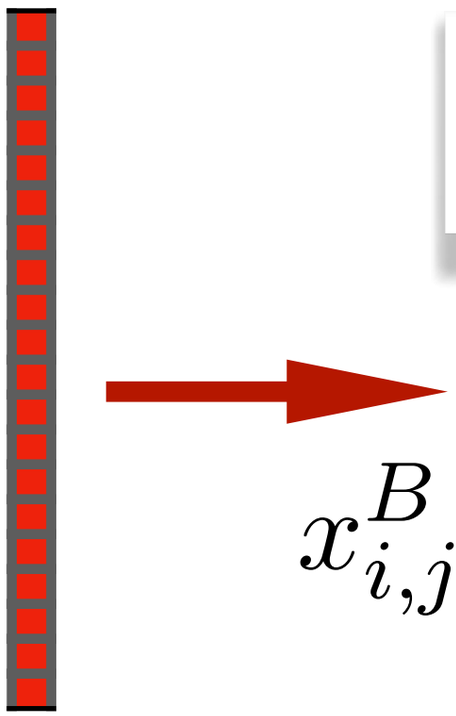
Features Binarization: Setup

Binarization Setup

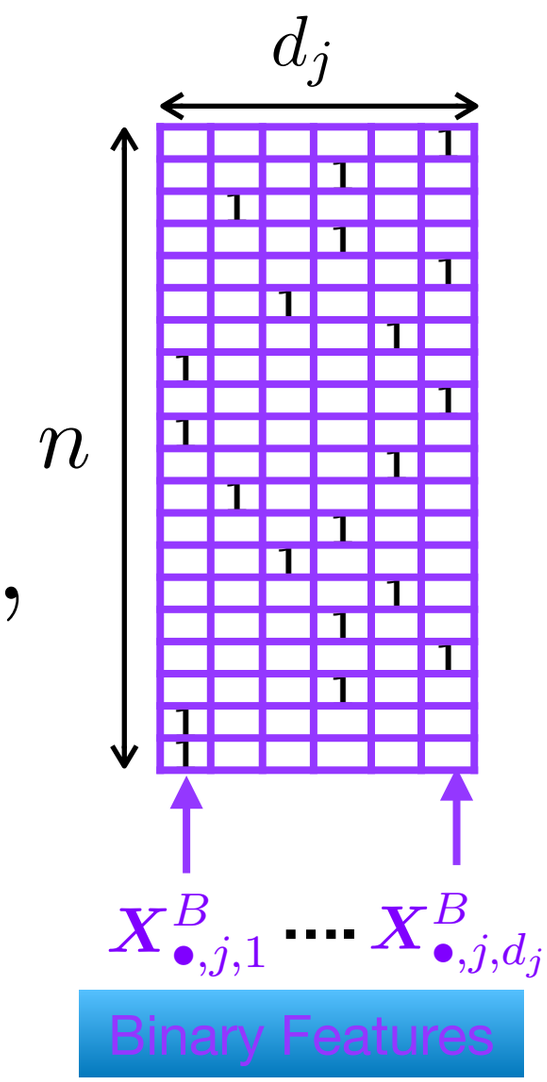
The j-th column $\boldsymbol{X}_{\bullet,j}$ is replaced by a number $d_j \geq 2$ of binary columns (containing only zeros and ones) $\boldsymbol{X}^B_{\bullet,j,1}, \ldots, \boldsymbol{X}^B_{\bullet,j,d_j}$

Partition of the range of values of $\boldsymbol{X}_{\bullet,j}$ into intervals $I_{j,1}, \ldots, I_{j,d_j}$ and put

$\boldsymbol{X}_{\bullet,j}$



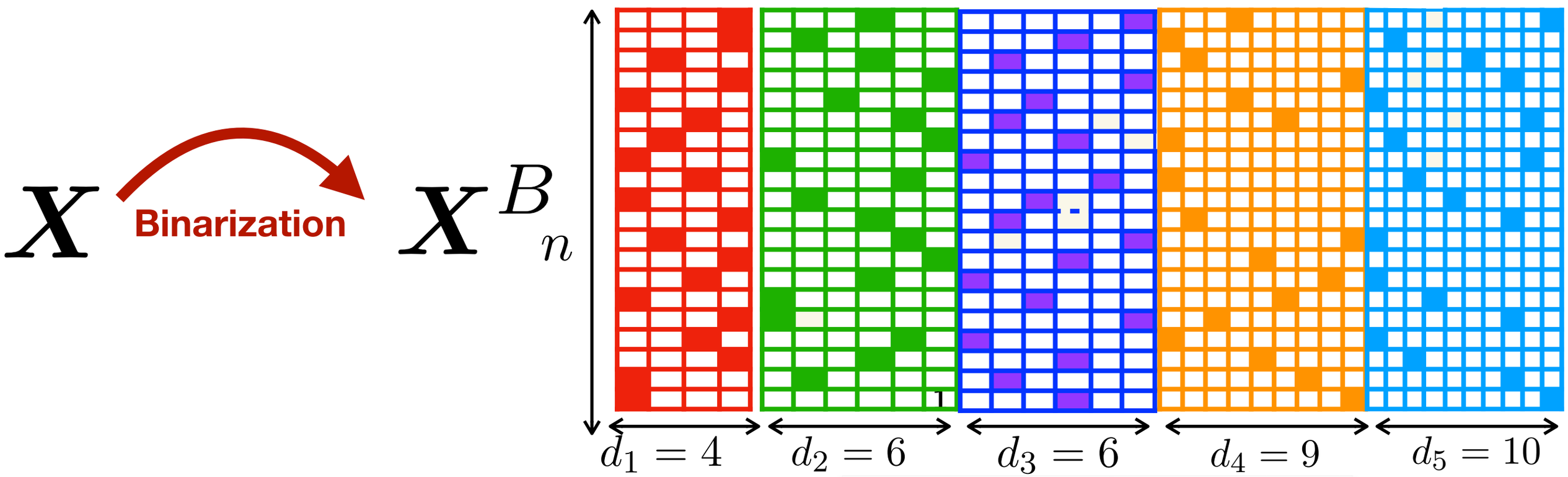
$$x^B_{i,j,k} = \begin{cases} 1, & \text{if } x_{i,j} \in I_{j,k}, \\ 0, & \text{otherwise} \end{cases}$$



Features Binarization: Setup

If $\boldsymbol{X}_{\bullet,j}$ takes values **(modalities)** in the set $\{1, \ldots, M_j\}$ with cardinality M_j, we take $d_j = M_j$ and use one-hot coding of each modality by defining:

$$x^B_{i,j,k} = \begin{cases} 1, & \text{if } x_{i,j} = k, \\ 0, & \text{otherwise} \end{cases}$$



Binarized Features Matrix

Features Binarization
Inter-quantile Partition Intervals

The i-th of the binarized matrix $\boldsymbol{X}^B$ reads as

$$x_i^B = [x_{i,1,1}^B, \ldots, x_{i,1,d_1}^B, x_{i,2,1}^B, \ldots, x_{i,2,d_2}^B, \ldots, x_{i,p,1}^B, \ldots, x_{i,p,d_p}^B]^\top \in \mathbb{R}^d$$

1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0

where

$$d = \sum_{j=1}^{p} d_j$$

Choice of the $I_{j,k}$ Intervals? → **Natural Choice: Inter-quantile intervals**

$$I_{j,1} = \left[q_j(0), q_j(\frac{1}{d_j})\right] \text{ and } I_{j,k} = \left(q_j(\frac{k-1}{d_j}), q_j(\frac{k}{d_j})\right]$$

for $k = 2, \ldots, d_j$, and where $q_j(\alpha)$ denotes a quantile of order $\alpha \in [0,1]$ for $\boldsymbol{X}_{\bullet,j}$

Features Binarization: Example

```python
import numpy as np
import pandas as pd
pd.option_context('display.max_rows', None,
                  'display.max_columns', None)
import prettytable
import seaborn as sns
# tick
from tick.preprocessing import FeaturesBinarizer

# Origin matrix
features = np.array([[0.00902084,  0.46519565, 'z'],
                     [0.46599565,  3.46523565,  2.],
                     [0.82091721,  -1.2650095,  2.],
                     [-0.17315496, 7.86545565,  1.],
                     [4.08180209,  6.26569565,  0.],
                     [1.6011727,   0.36548565,  0.],
                     [2.7347947,   1.46500565,  20.],
                     [-5.9890938,  4.55529565,  0.],
                     [6.3063761,   2.22548565,  1.],
                     [9.27110903,  -3.46514565, 0.]])

df = pd.DataFrame(data=features)

# Binarization preprocessing with $d_j=4$
# for continuous features
binarizer = FeaturesBinarizer(n_cuts=3)
binarized_features = binarizer.fit_transform(features)

# binarized matrix X^B
X_bin = binarized_features.toarray()
# print(X_bin.shape) # (10, 13)

columns_bin =
 ["$X^B_{.11}$", "$X^B_{.12}$", "$X^B_{.13}$", "$X^B_{.14}$",
  "$X^B_{.21}$", "$X^B_{.22}$", "$X^B_{.23}$", "$X^B_{.24}$",
  "$X^B_{.31}$", "$X^B_{.32}$", "$X^B_{.33}$", "$X^B_{.34}$",
  "$X^B_{.35}$"]
df_bin = pd.DataFrame(data=X_bin, columns=columns_bin)
pd.options.display.float_format = '{:,.0f}'.format

# plot
cm = sns.light_palette("purple", as_cmap=True)
(df_bin.style
  .background_gradient(cmap=cm)
  # .highlight_max(subset=['total_amt_usd_diff','total_amt_usd_pct_diff'])
  .set_caption('${\qquad \qquad \qquad  \Huge{Binarized Features Matrix}}$')
  .format({'total_amt_usd_pct_diff': "{:.2%}"}))
```

tick
ML /Python Package

(Bacry et al., 2018)

Binarized Features Matrix

$X^B_{.11}$	$X^B_{.12}$	$X^B_{.13}$	$X^B_{.14}$	$X^B_{.21}$	$X^B_{.22}$	$X^B_{.23}$	$X^B_{.24}$	$X^B_{.31}$	$X^B_{.32}$	$X^B_{.33}$	$X^B_{.34}$	$X^B_{.35}$
1	0	0	0	0	1	0	0	0	0	0	0	1
0	1	0	0	0	0	1	0	0	0	1	0	0
0	1	0	0	1	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	1	0	1	0	0	0
0	0	1	0	0	0	0	1	1	0	0	0	0
0	0	1	0	1	0	0	0	1	0	0	0	0
0	0	1	0	0	1	0	0	0	0	0	1	0
1	0	0	0	0	0	1	0	1	0	0	0	0
0	0	0	1	0	0	1	0	0	1	0	0	0
0	0	0	1	1	0	0	0	1	0	0	0	0

Features Binarization: Weights

Weights of one-hot encoded features
To each binarized feature $\boldsymbol{X}^B_{\bullet,j,k}$ corresponds a parameter $\theta_{j,k}$

The parameters associated to the binarization of the j-th feature is denoted

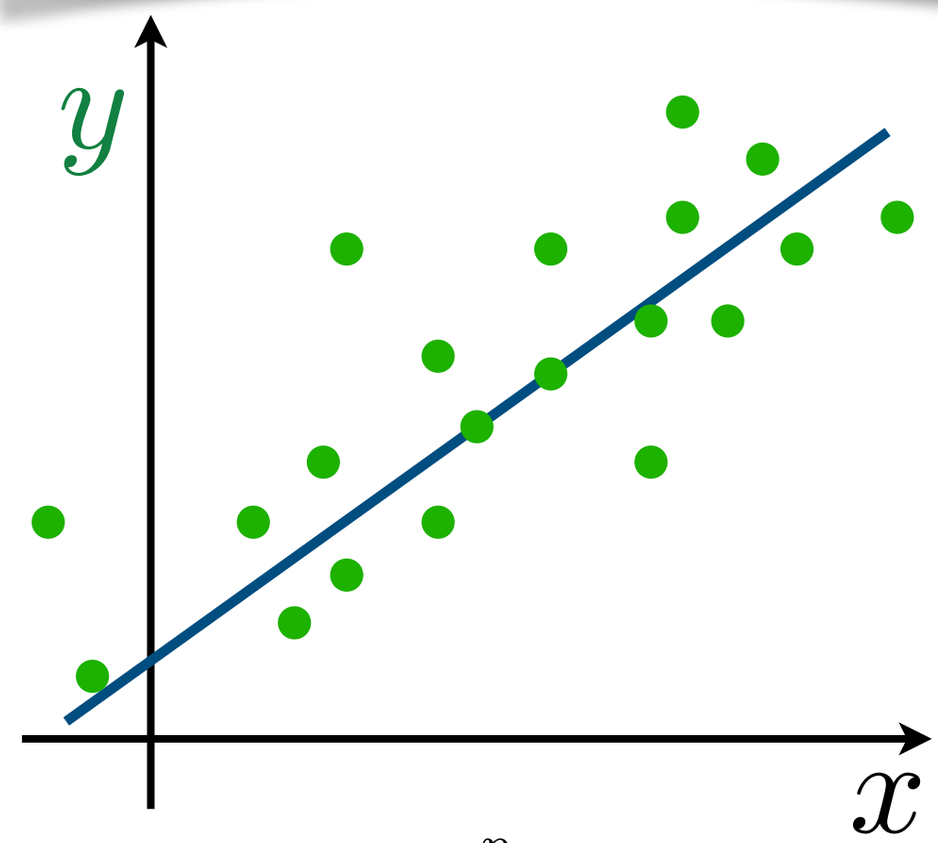
$$\theta_{j,\bullet} = [\theta_{j,1} \ \cdots \ \theta_{j,d_j}]^\top$$

The full parameters vector of size $d = \sum_{j=1}^{p} d_j$, is simply

$$\theta = [\theta_{1,1} \ \cdots \ \theta_{1,d_1} \ \theta_{2,1} \ \cdots \ \theta_{2,d_2} \ \cdots \ \theta_{p,1} \ \cdots \ \theta_{p,d_p}]^\top \in \mathbb{R}^d$$

Features Binarization: Weights

Linear regression on raw features

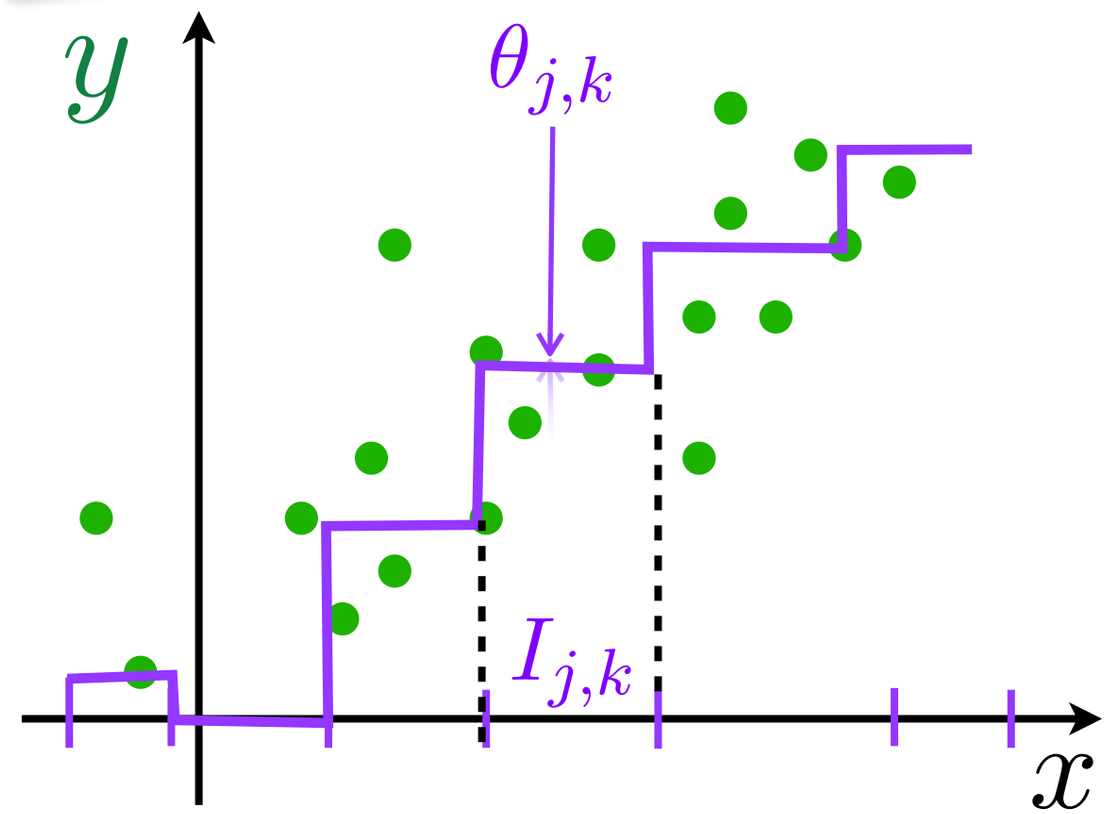


$$y_i = \omega^\top x_i + b = \sum_{j=1}^{p} \omega_j x_{i,j} + b$$

*Impact of the j-th feature is **linear** and encoded by a **single weight** ω_j*

$$x \mapsto \omega_j x$$

Linear regression on binarized features



$$y_i = \sum_{j=1}^{p} \sum_{k=1}^{d_j} \theta_{j,k} x_{i,j,k} + b = \sum_{j=1}^{p} \sum_{k=1}^{d_j} \theta_{j,k} \mathbb{1}(x_{i,j} \in I_{j,k}) + b$$

*Impact of the j-th feature is **piecewise constant** and encoded by a **block***

$$\theta_{j,\bullet} = [\theta_{j,1} \cdots \theta_{j,d_j}]^\top$$

$$x \mapsto \sum_{k=1}^{d_j} \theta_{j,k} \mathbb{1}(x \in I_{j,k})$$

Features Binarization: Issues

(P1) Colinear binary features
One-hot-encodings satisfy

$$\sum_{k=1}^{d_j} x^B_{i,j,k} = 1, \text{ for all } j = 1, \ldots, p$$

$$X^B$$

Not full rank

(P2) Overparametrization
Increasing the number of bins d_j

Overfitting

(P3) Feature selection
Some of the raw features $X_{\bullet,j}$ might be not relevant for the prediction task!

Block-Sparsity!

$$\theta_{j,1} = 0, \ldots, \theta_{j,d_j} = 0$$

Features Binarization: Solutions

To deal with **(P1)**, we impose a **linear constraint** in each block (Agresti, 2015)

(S1) $$n_j^\top \theta_{j,\bullet} = \sum_{k=1}^{d_j} n_{j,k}\theta_{j,k} = 0 \text{ for all } j = 1, \ldots, p$$

$$n_j = [n_{j,1}, \ldots, n_{j,d_j}]^\top \in \mathbb{N}^{d_j} \quad \text{where} \quad n_{j,k} = |\{i : x_{i,j} \in I_{j,k}\}|$$

To tackle **(P2)**, we keep the number of different values taken by $\theta_{j,\bullet}$ to minimal level by using a within **block weighted total-variation penalization**

(S2) $$\sum_{j=1}^{p} \|\theta_{j,\bullet}\|_{\mathrm{TV},\hat{\omega}_{j,\bullet}} = \sum_{k=2}^{d_j} \hat{\omega}_{j,k}|\theta_{j,k} - \theta_{j,k-1}|$$

(S1) + (S2) solve **(P3)**

Binarsity

$$\mathrm{bina}(\theta) = \sum_{j=1}^{p} \left(\sum_{k=2}^{d_j} \hat{\omega}_{j,k} |\theta_{j,k} - \theta_{j,k-1}| + \delta_j(\theta_{j,\bullet}) \right)$$

where $$\delta_j(u) = \begin{cases} 0 & \text{if} \quad n_j^\top u = 0, \\ \infty & \text{otherwise} \end{cases}$$

and $$\hat{\omega}_{j,k} = \mathcal{O}\left(\sqrt{\frac{\log d}{n} \hat{\pi}_{j,k}} \right)$$ with $$\hat{\pi}_{j,k} = \frac{\left| \left\{ i = 1, \ldots, n : x_{i,j} \in \cup_{k'=k}^{d_j} I_{j,k'} \right\} \right|}{n}$$

$\hat{\pi}_{j,k}$ = Proportion of **1**'s in the sub-matrix obtained by deleting the first k columns in the j-th binarized block matrix

Binarsity: Interpretation

Large d_j → $\theta_{j,k}$ and $\theta_{j,k-1}$ to be close

Penalization ℓ_1 → **Sparsity**

Penalization $\boldsymbol{Binarsity}$ → **Piecewise constant** $\theta_{j,\bullet}$

If $\theta_{j,\bullet}$ is constant than the linear constraint ${n_j}^\top \theta_{j,\bullet} = 0$ entails $\theta_{j,\bullet} \equiv 0$

Block-Sparsity

$\boldsymbol{X}_{\bullet,j}$

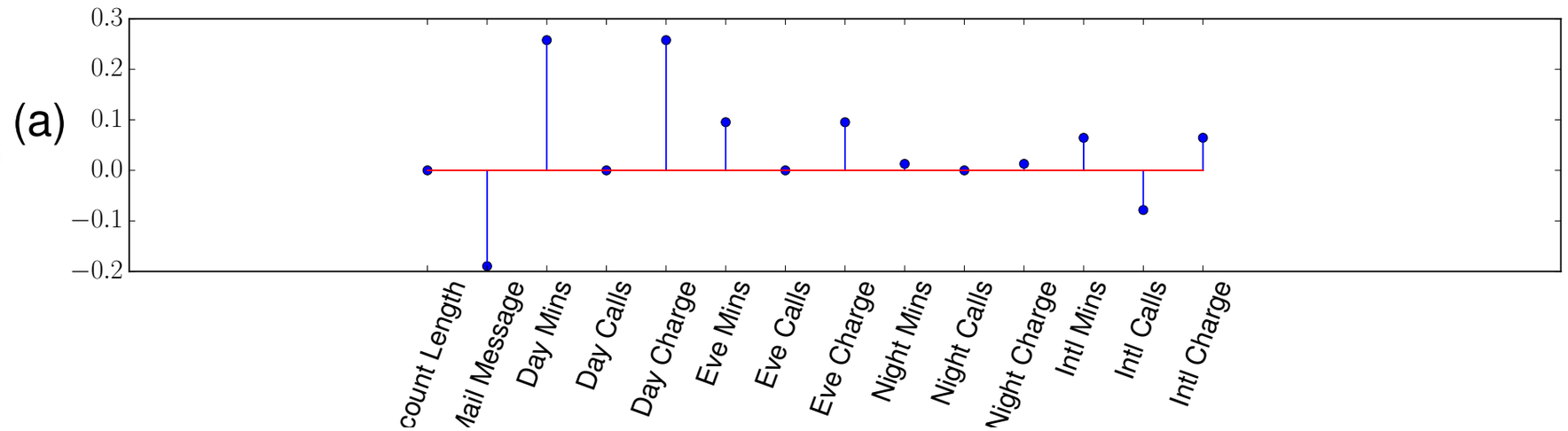
Statistically relevant if $\theta_{j,\bullet} \not\equiv 0$

Statistically not relevant if $\theta_{j,\bullet} \equiv 0$

Illustrations

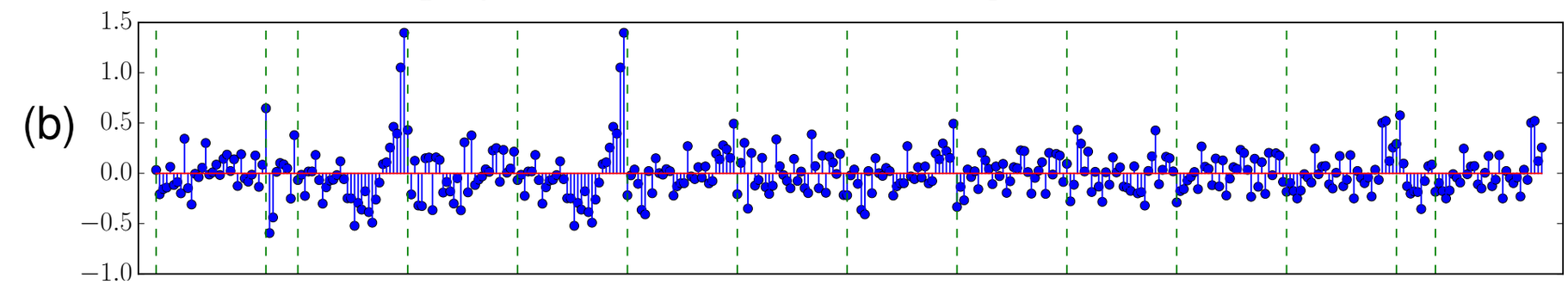
Weights of a logistic Regression on Churn Dataset (UCI) $n = 3333, p = 14$

Raw continuous features



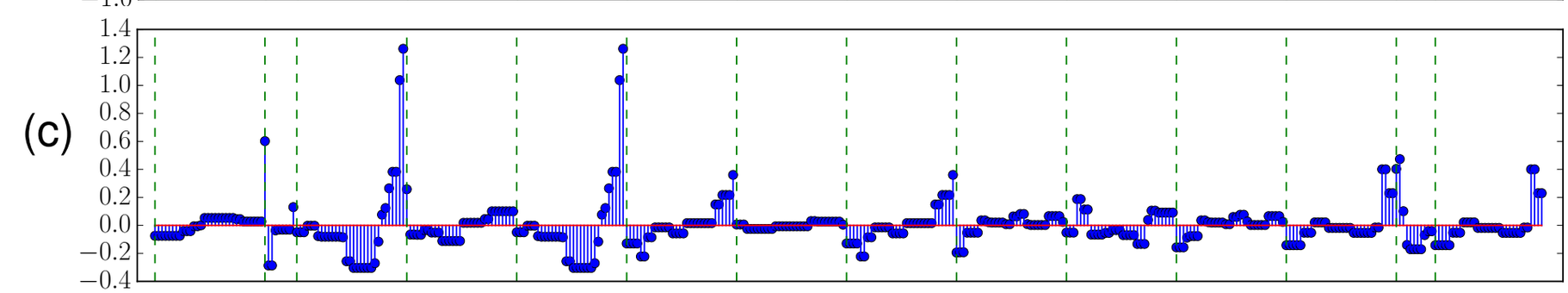
Binarized features

No-penalization



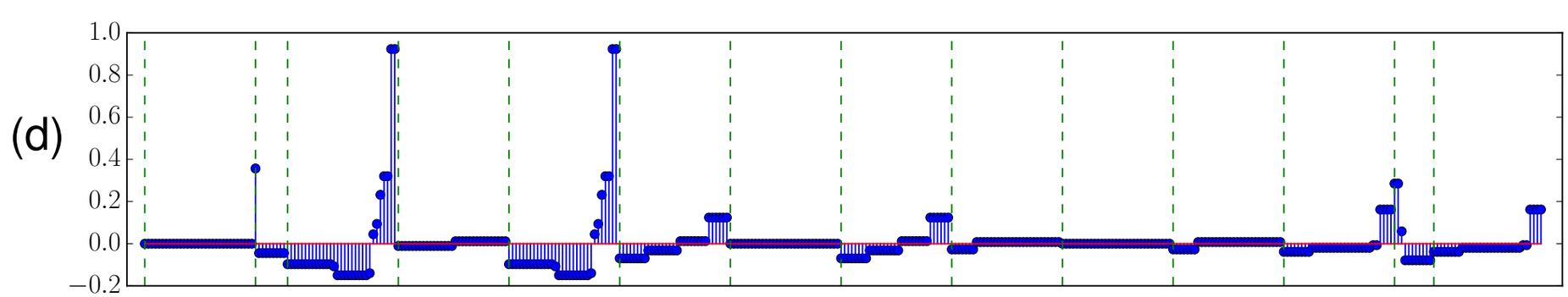
Binarized features

Low binarsity penalization



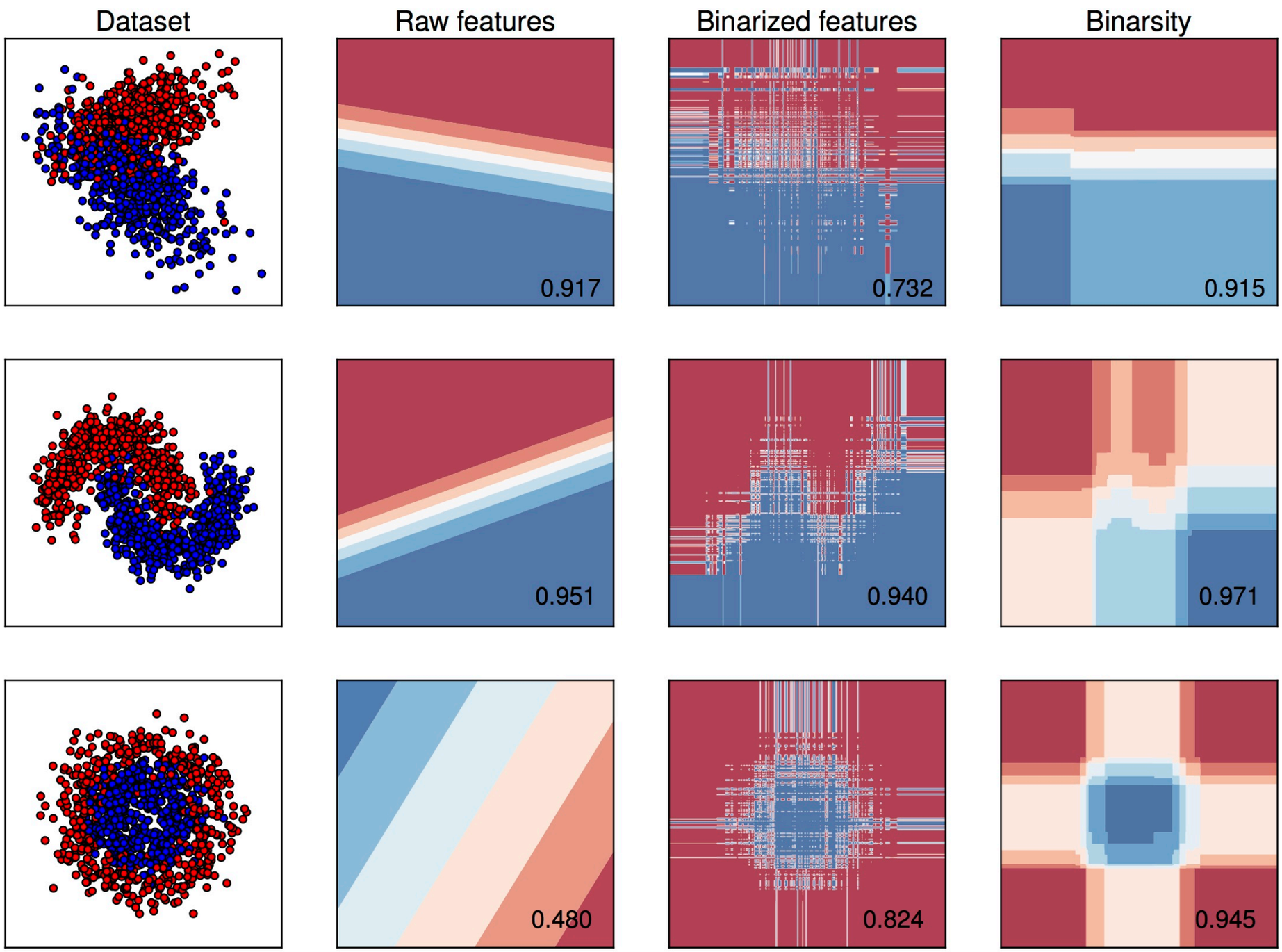
Binarized features

Strong binarsity penalization



Decision Boundaries of a Logistic Regression

Toy datasets with $n = 1000, p = 2$ and $d_1 = d_2 = 100$



Theoretical Guarantees (GLM + Binarsity)

Generalized Linear Models

$$\mathbb{P}(y|x) = \exp\left(\frac{y m^0(x) - b(m^0(x))}{\phi} + c(y, \phi)\right)$$

The functions $b(\cdot)$ and $c(\cdot)$ and the dispersion parameter ϕ are **known**.

The natural parameter $m^0(\cdot)$ is **unknown** with

$$m^0(x) = g(\mathbb{E}[y|x]), \text{ where } b' = g^{-1}$$

Examples:
Logistic and probit regression for binary data or multinomial regression for categorical data, Poisson regression for count data, etc ...

GLM: Goodness-of-fit

Empirical risk

$$R_n(m_\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, m_\theta(x_i))$$

where

$$m_\theta(x) = \theta^\top x^B$$

GLM loss function $\ell(y, y') = -yy' + b(y')$

We estimate m^0 by $\hat{m} = m_{\hat{\theta}}$

$$\hat{\theta} \in \operatorname*{argmin}_{\theta \in \mathbb{R}^d} \left\{ R_n(m_\theta) + \mathrm{bina}(\theta) \right\}$$

Fast Oracle Inequality for GLM + Binarsity

Assumptions on GLM

$$\mathbb{P}(y|x) = \exp\left(\frac{y m^0(x) - b(m^0(x))}{\phi} + c(y,\phi)\right)$$

A1. $b(\cdot)$ is three times continuously

A2. $|b'''(z)| \leq C_b |b''(z)|$ for some $C_b > 0$

A3. $C_n = \max_{i=1,\ldots,n} |m^0(x_i)| < \infty$

A4. $L_n \leq \max_{i=1,\ldots,n} b''(m^0(x_i)) \leq U_n$

Satisfied for the following standard GLM :

Model	ϕ	$b(z)$	$b'(z)$	$b''(z)$	$b'''(z)$	C_b	L_n	U_n
Normal	σ^2	$\frac{z^2}{2}$	z	1	0	0	1	1
Logistic	1	$\log(1+e^z)$	$\frac{e^z}{1+e^z}$	$\frac{e^z}{(1+e^z)^2}$	$\frac{1-e^z}{1+e^z} b''(z)$	2	$\frac{e^{C_n}}{(1+e^{C_n})^2}$	1/4
Poisson	1	e^z	e^z	e^z	e^z	1	e^{-C_n}	e^{C_n}

Fast Oracle Inequality for GLM + Binarsity

Non asymptotic oracle inequality in terms of **Excess risk**!

$$R(\hat{m}) - R(m_0) = \mathbb{E}_{\mathbb{P}(y|x)}[R_n(\hat{m}) - R_n(m_0)]$$

A new measures of sparsity: binarsity

For $\theta \in \mathbb{R}^d$, let $J(\theta) = \big[J_1(\theta), \ldots, J_p(\theta)\big]$ be the concatenation of the support sets relative to the total-variation penalization, that is

$$J_j(\theta) = \big\{k : \theta_{j,k} \neq \theta_{j,k-1}, \text{ for } k = 2, \ldots, d_j\big\}.$$

$$\boldsymbol{binarsity}(\theta) = |J(\theta)| = \sum_{j=1}^{p} |J_j(\theta)| = \sum_{j=1}^{p} \big|\big\{k : \theta_{j,k} \neq \theta_{j,k-1}, \text{ for } k = 2, \ldots, d_j\big\}\big|$$

$\boldsymbol{binarsity}(\theta)$ **Counts the number of non-equal consecutive values of** θ

Fast Oracle Inequality for GLM + Binarsity

Restricted Eigenvalues Assumption

Let $K = [K_1, \ldots, K_p]$ be a concatenation of index sets such that $\sum_{j=1}^p |K_j| \leq J^\star$. Assume

$$\kappa(K) \in \inf_{u \in \mathscr{C}_{\mathrm{TV},\hat{\omega}}(K) \setminus \{\mathbf{0}_d\}} \left\{ \frac{\|\boldsymbol{X}^B u\|_2}{\sqrt{n}\|u_K\|_2} \right\} > 0$$

with $\mathscr{C}_{\mathrm{TV},\hat{\omega}}(K) = \left\{ u \in \mathbb{R}^d : \sum_{j=1}^p \|(u_{j,\bullet})_{K_j^\complement}\|_{\mathrm{TV},\hat{\omega}_{j,\bullet}} \leq 2 \sum_{j=1}^p \|(u_{j,\bullet})_{K_j}\|_{\mathrm{TV},\hat{\omega}_{j,\bullet}} \right\}$.

Theorem: Fast Rate $\approx \big(\boldsymbol{binarsity}(\theta) \times \log(d)\big)/n$

$$R(m_{\hat{\theta}}) - R(m^0) \leq \inf_{\substack{\theta \in B_d(\rho) \\ \forall j\ \mathbf{1}^\top \theta_{j,\bullet} = 0 \\ |J(\theta)| \leq J^\star}} \Bigg\{ 3(R(m_\theta) - R(m^0)) + \frac{\xi |J(\theta)|}{\kappa^2(J(\theta))} \max_{j=1,\ldots,p} \|(\hat{\omega}_{j,\bullet})_{J_j(\theta)}\|_\infty^2 \Bigg\},$$

where $B_d(\rho) = \{\theta \in \mathbb{R}^d : \sum_{j=1}^p \|\theta_{j,\bullet}\|_\infty \leq \rho\}$, and $\xi = Cst(C_n, \rho, p, L_n, U_n)$.

Implementations

Optimization problem

Need to optimize

$$R_n(\theta) + \mathrm{bina}(\theta)$$

Smooth (Gradient Lipschitz) **Non-differentiable**

First order optimization: Proximal Gradient Descent (Bach et al., 2012)

$$\theta^{(t+1)} \leftarrow \mathrm{prox}_{\eta\,\mathrm{bina}(\cdot)}\left(\theta^{(t)} - \eta \nabla R_n(\theta^{(t)})\right)$$

proximal operator

learning rate

$$\mathrm{prox}_{\lambda g}(y) = \underset{\theta \in \mathbb{R}^n}{\mathrm{argmin}} \left\{ \frac{1}{2}\|y - \theta\|_2^2 + \lambda g(\theta) \right\}, \text{ for all } y \in \mathbb{R}^n.$$

Proximal Operator of Binarsity

Binarsity is separable by blocks, then for all j=1, ..p,

$$\left(\operatorname{prox}_{\text{bina}}(\theta)\right)_{j,\bullet} = \operatorname{prox}_{(\|\cdot\|_{\text{TV},\hat{\omega}_{j,\bullet}}+\delta_j)}(\theta_{j,\bullet})$$

The following algorithm expresses $\operatorname{prox}_{\text{bina}}$

Algorithm 1:

Input: vector $\theta \in \mathbb{R}^d$, weights $\hat{\omega}_{j,k}$ and $n_{j,k}$ for $j = 1, \ldots, p$ and $k = 1, \ldots, d_j$

Output: vector $\eta = \operatorname{prox}_{\text{bina}}(\theta)$

for $j = 1$ ***to*** p **do**

- $\beta_{j,\bullet} \leftarrow \operatorname{prox}_{\|\theta_{j,\bullet}\|_{\text{TV},\hat{\omega}_{j,\bullet}}}(\theta_{j,\bullet})$ (weighted TV penalization in block $\theta_{j,\bullet}$)
- $\eta_{j,\bullet} \leftarrow \beta_{j,\bullet} - \frac{n_j \top \beta_{j,\bullet}}{\|n_j\|_2^2} n_j$ (project onto $\operatorname{span}(n_j)^{\perp}$)

Return: η

Proximal Operator of Weighted TV

$$\hat{\theta} = \mathrm{prox}_{\|\cdot\|_{\mathrm{TV},\hat{\omega}}}(y) = \underset{\theta \in \mathbb{R}^n}{\mathrm{argmin}} \left\{ \frac{1}{2}\|y-\theta\|_2^2 + \|\theta\|_{\mathrm{TV},\hat{\omega}} \right\}$$

$\hat{\theta} = \mathrm{prox}_{\|\cdot\|_{\mathrm{TV},\hat{\omega}}}(y)$ [Alaya et al.'15]

1. **set** $k = k_0 = k_- = k_+ \leftarrow 1$; $\theta_{\min} \leftarrow y_1 - \hat{\omega}_2$; $\theta_{\max} \leftarrow y_1 + \hat{\omega}_2$; $u_{\min} \leftarrow \hat{\omega}_2$; $u_{\max} \leftarrow -\hat{\omega}_2$;
2. **if** $k = n$ **then**
   - $\hat{\theta}_n \leftarrow \theta_{\min} + u_{\min}$;
3. **if** $y_{k+1} + u_{\min} < \theta_{\min} - \hat{\omega}_{k+2}$ **then** /* negative jump */
   - $\hat{\theta}_{k_0} = \cdots = \hat{\theta}_{k_-} \leftarrow \theta_{\min}$; $k = k_0 = k_- = k_+ \leftarrow k_- + 1$;
   - $\theta_{\min} \leftarrow y_k - \hat{\omega}_{k+1} + \hat{\omega}_k$; $\theta_{\max} \leftarrow y_k + \hat{\omega}_{k+1} + \hat{\omega}_k$; $u_{\min} \leftarrow \hat{\omega}_{k+1}$; $u_{\max} \leftarrow -\hat{\omega}_{k+1}$;
4. **else if** $y_{k+1} + u_{\max} > \theta_{\max} + \hat{\omega}_{k+2}$ **then** /* positive jump */
   - $\hat{\theta}_{k_0} = \ldots = \hat{\theta}_{k_+} \leftarrow \theta_{\max}$; $k = k_0 = k_- = k_+ \leftarrow k_+ + 1$;
   - $\theta_{\min} \leftarrow y_k - \hat{\omega}_{k+1} - \hat{\omega}_k$; $\theta_{\max} \leftarrow y_k + \hat{\omega}_{k+1} - \hat{\omega}_k$; $u_{\min} \leftarrow \hat{\omega}_{k+1}$; $u_{\max} \leftarrow -\hat{\omega}_{k+1}$;
5. **else** /* no jump */
   - **set** $k \leftarrow k+1$; $u_{\min} \leftarrow y_k + \hat{\omega}_{k+1} - \theta_{\min}$; $u_{\max} \leftarrow y_k - \hat{\omega}_{k+1} - \theta_{\max}$;
   - **if** $u_{\min} \geq \hat{\omega}_{k+1}$ **then**
     - $\theta_{\min} \leftarrow \theta_{\min} + \frac{u_{\min} - \hat{\omega}_{k+1}}{k - k_0 + 1}$; $u_{\min} \leftarrow \hat{\omega}_{k+1}$; $k_- \leftarrow k$;
   - **if** $u_{\max} \leq -\hat{\omega}_{k+1}$ **then**
     - $\theta_{\max} \leftarrow \theta_{\max} + \frac{u_{\max} + \hat{\omega}_{k+1}}{k - k_0 + 1}$; $u_{\max} \leftarrow -\hat{\omega}_{k+1}$; $k_+ \leftarrow k$;
6. **if** $k < n$ **then**
   - **go to 3.**;
7. **if** $u_{\min} < 0$ **then**
   - $\hat{\theta}_{k_0} = \cdots = \hat{\theta}_{k_-} \leftarrow \theta_{\min}$; $k = k_0 = k_- \leftarrow k_- + 1$; $\theta_{\min} \leftarrow y_k - \hat{\omega}_{k+1} + \hat{\omega}_k$;
   - $u_{\min} \leftarrow \hat{\omega}_{k+1}$; $u_{\max} \leftarrow y_k + \hat{\omega}_k - v_{\max}$; **go to 2.**;
8. **else if** $u_{\max} > 0$ **then**
   - $\hat{\theta}_{k_0} = \cdots = \hat{\theta}_{k_+} \leftarrow \theta_{\max}$; $k = k_0 = k_+ \leftarrow k_+ + 1$; $\theta_{\max} \leftarrow y_k + \hat{\omega}_{k+1} - \hat{\omega}_k$;
   - $u_{\max} \leftarrow -\hat{\omega}_{k+1}$; $u_{\min} \leftarrow y_k - \hat{\omega}_k - u_{\min}$; **go to 2.**;
9. **else**
   - $\hat{\theta}_{k_0} = \cdots = \hat{\theta}_n \leftarrow \theta_{\min} + \frac{u_{\min}}{k - k_0 + 1}$;

Numerical Experiments

Binary Classification

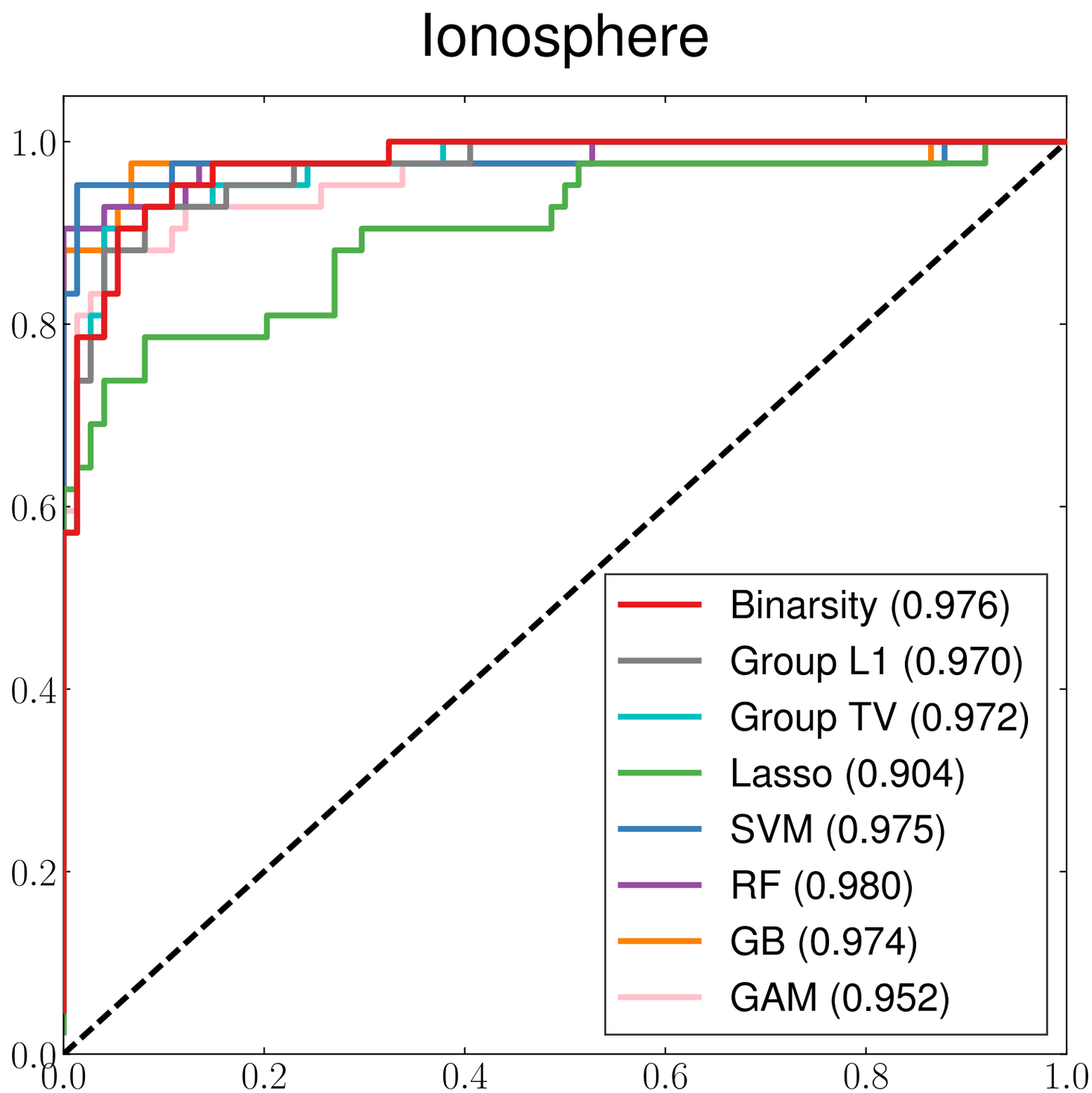
Source: UCI Machine Learning Repository

Dataset	#Samples	#Features
Ionosphere	351	34
Churn	3333	21
Default of Credit card	30000	24
Adult	32561	14
Bank Marketing	45211	17
Covertype	550088	10
SUSY	5000000	18
HEPMASS	10500000	28
HIGGS	11000000	24

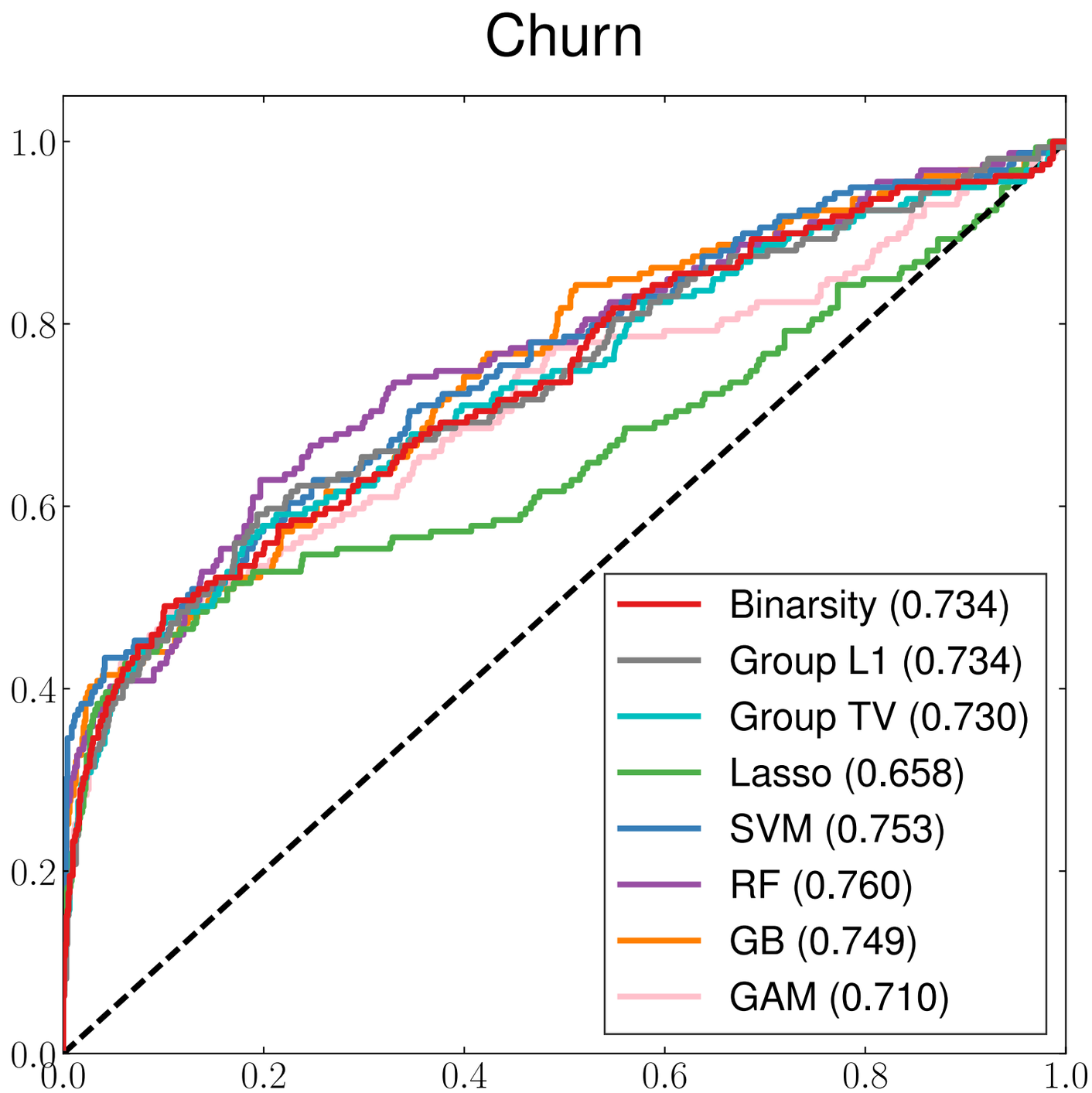
Baselines

Method	Description
Lasso	Logistic regression with ℓ_1 penalization
Group Lasso	Logistic regression with group ℓ_1 penalization
Group TV	Logistic regression with Group Total-Variation penalization
SVM	Support Vector Machine with Gaussian kernel
GAM	Generalized Additive Model
RF	Random Forest
GB	Gradient Boosting

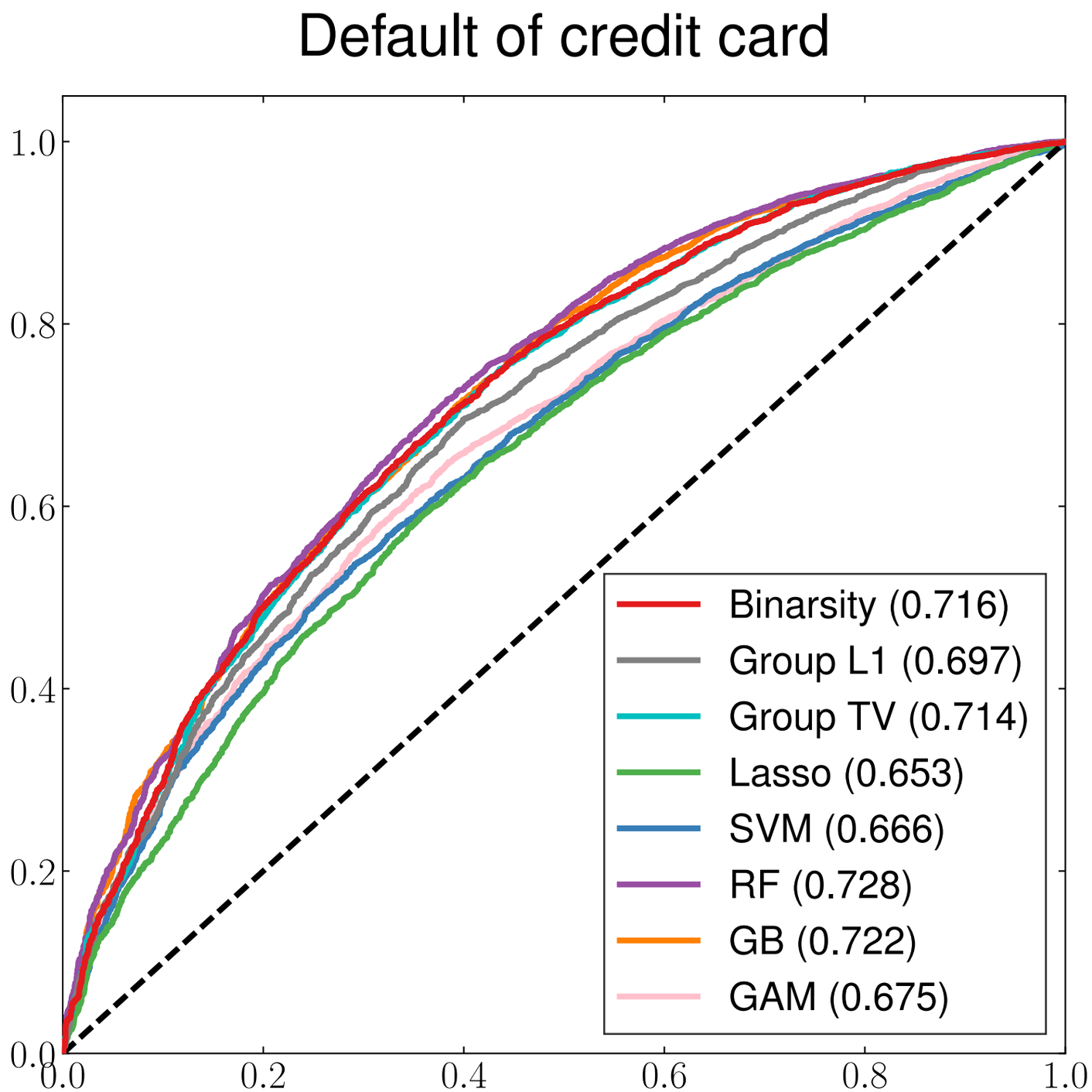
Results


Ionosphere
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Binarsity (0.976)
Group L1 (0.970)
Group TV (0.972)
Lasso (0.904)
SVM (0.975)
RF (0.980)
GB (0.974)
GAM (0.952)

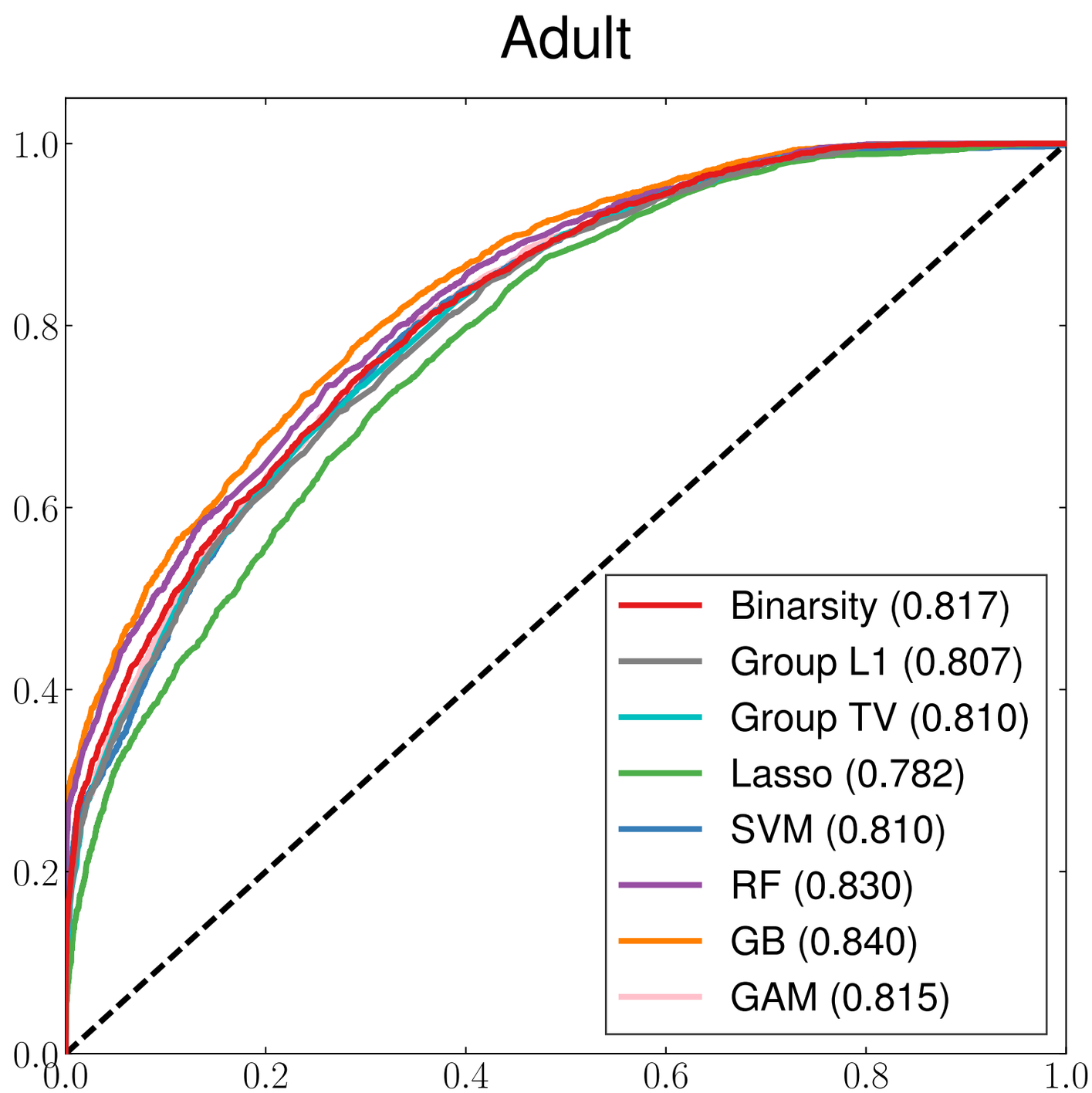
Results


Churn
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Binarsity (0.734)
Group L1 (0.734)
Group TV (0.730)
Lasso (0.658)
SVM (0.753)
RF (0.760)
GB (0.749)
GAM (0.710)

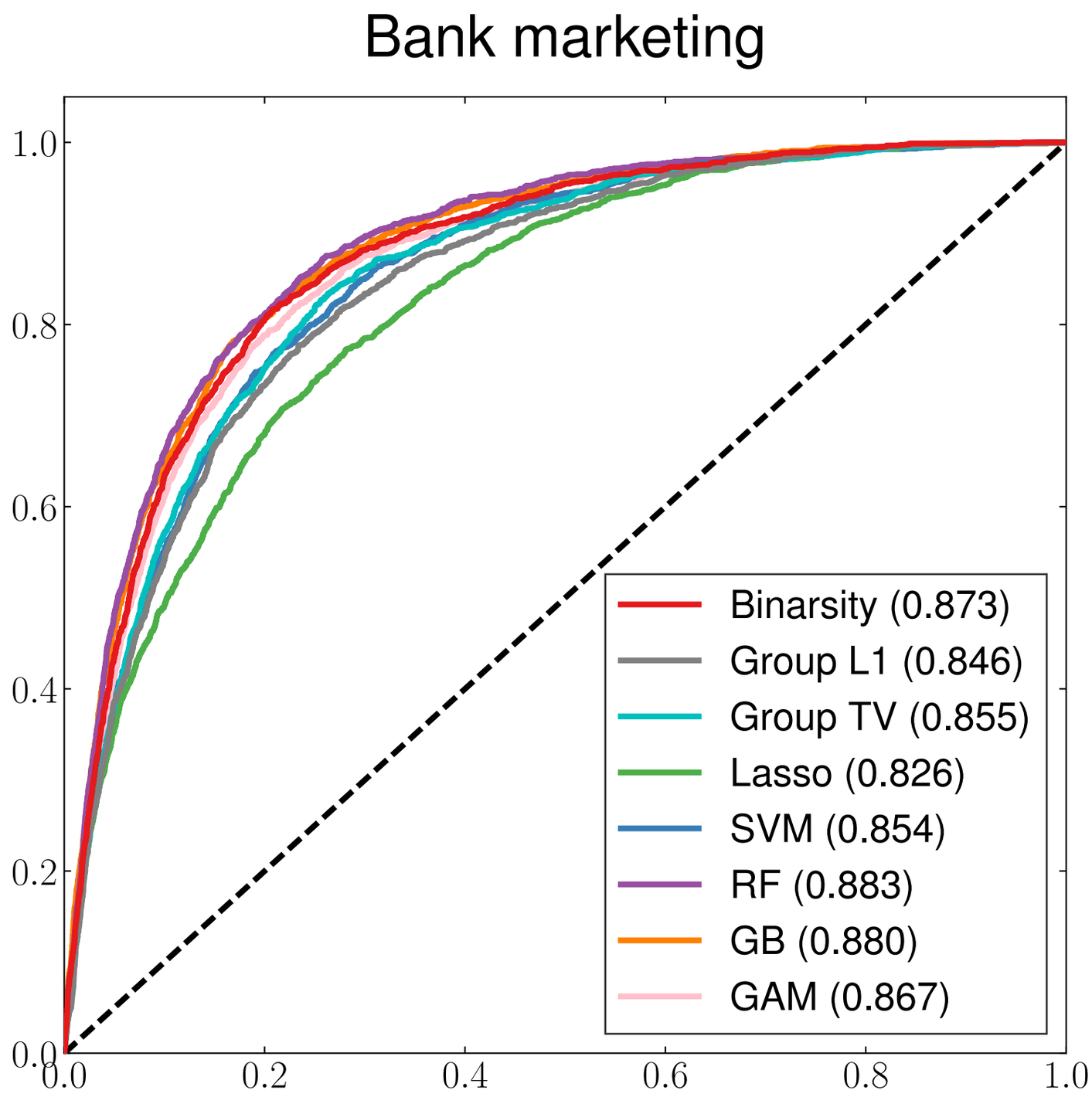
Results


Default of credit card
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Binarsity (0.716)
Group L1 (0.697)
Group TV (0.714)
Lasso (0.653)
SVM (0.666)
RF (0.728)
GB (0.722)
GAM (0.675)

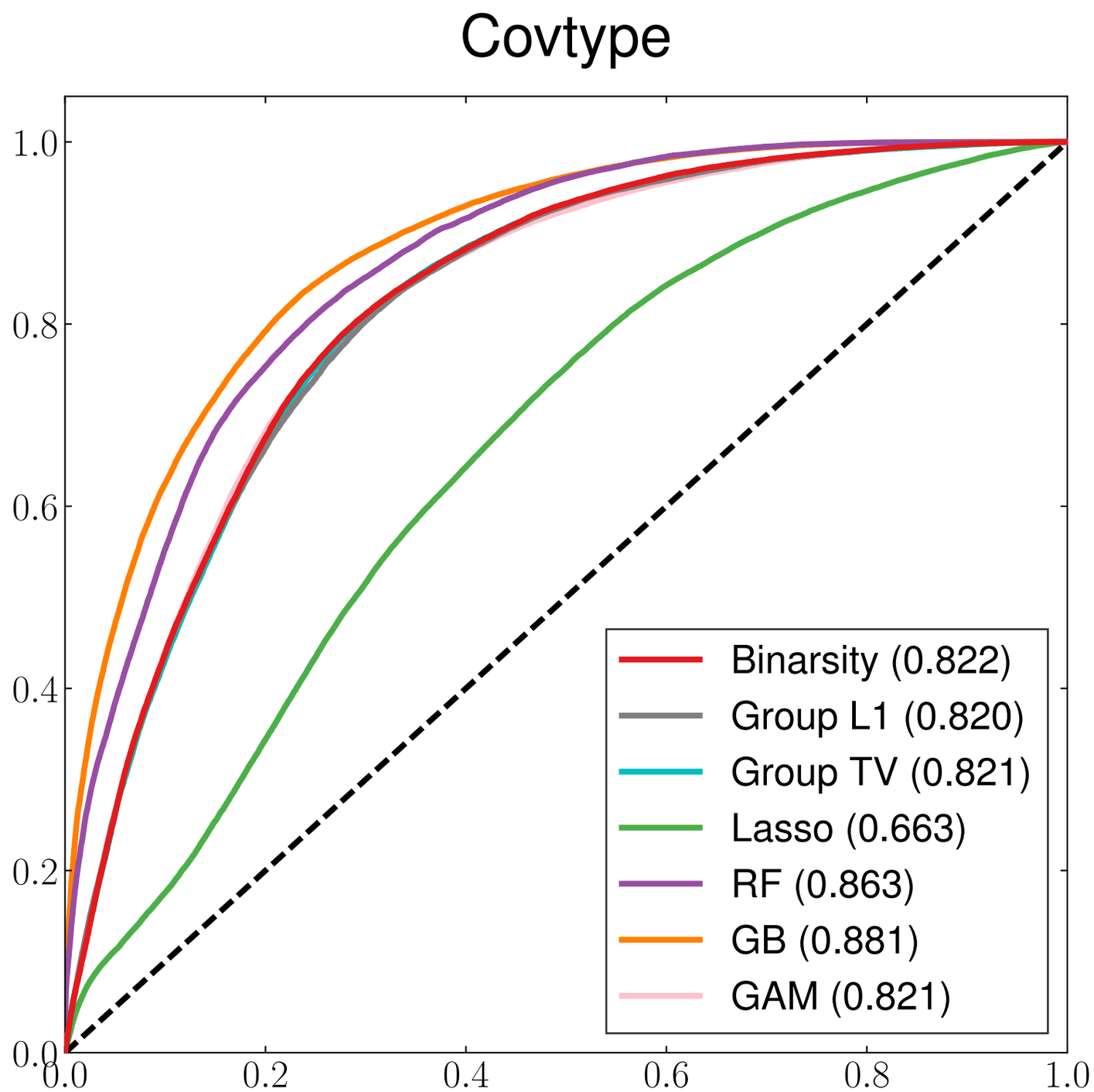
Results


Adult
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Binarsity (0.817)
Group L1 (0.807)
Group TV (0.810)
Lasso (0.782)
SVM (0.810)
RF (0.830)
GB (0.840)
GAM (0.815)

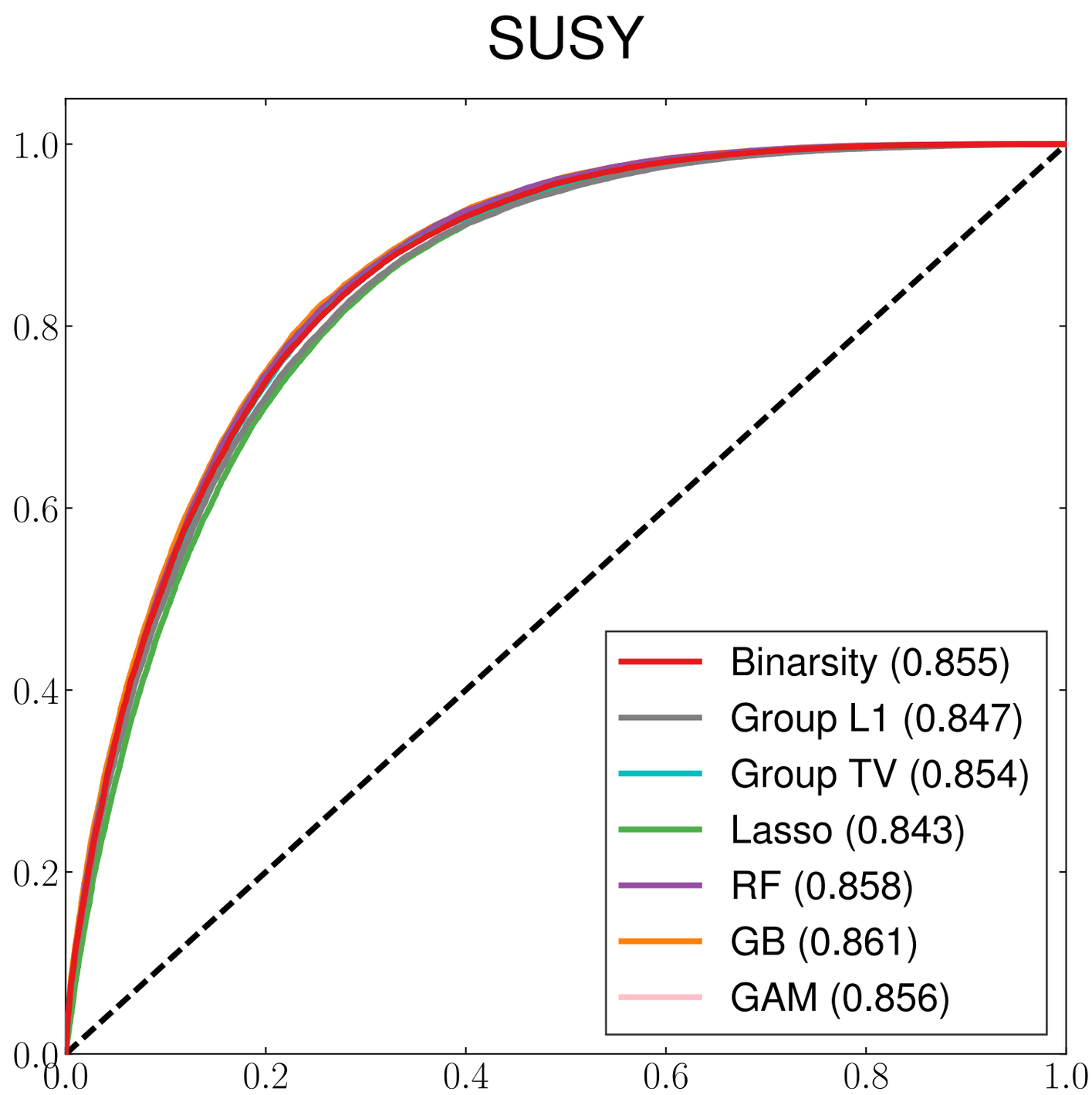
Results


Bank marketing
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Binarsity (0.873)
Group L1 (0.846)
Group TV (0.855)
Lasso (0.826)
SVM (0.854)
RF (0.883)
GB (0.880)
GAM (0.867)

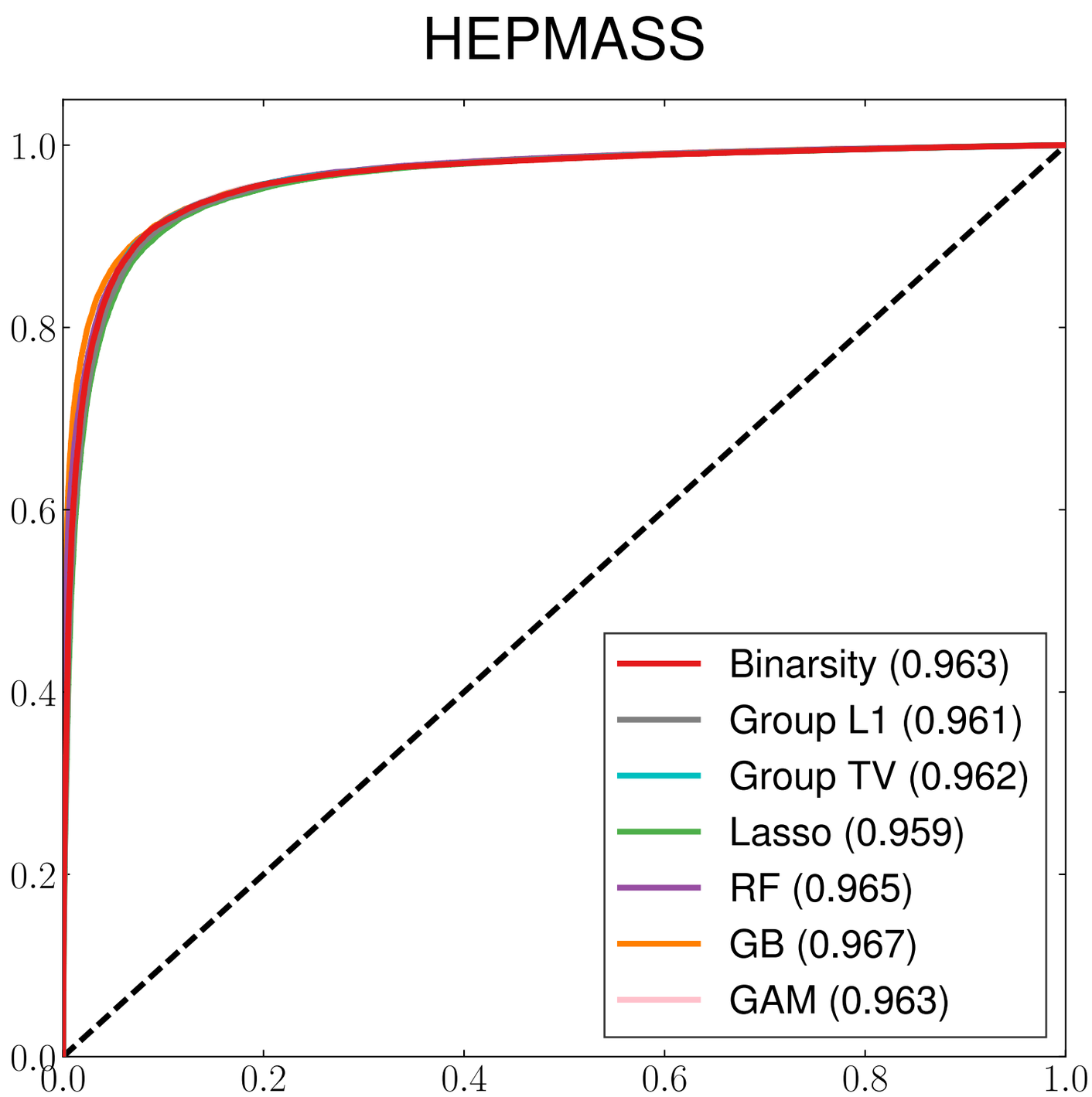
Results


Covtype
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Binarsity (0.822)
Group L1 (0.820)
Group TV (0.821)
Lasso (0.663)
RF (0.863)
GB (0.881)
GAM (0.821)

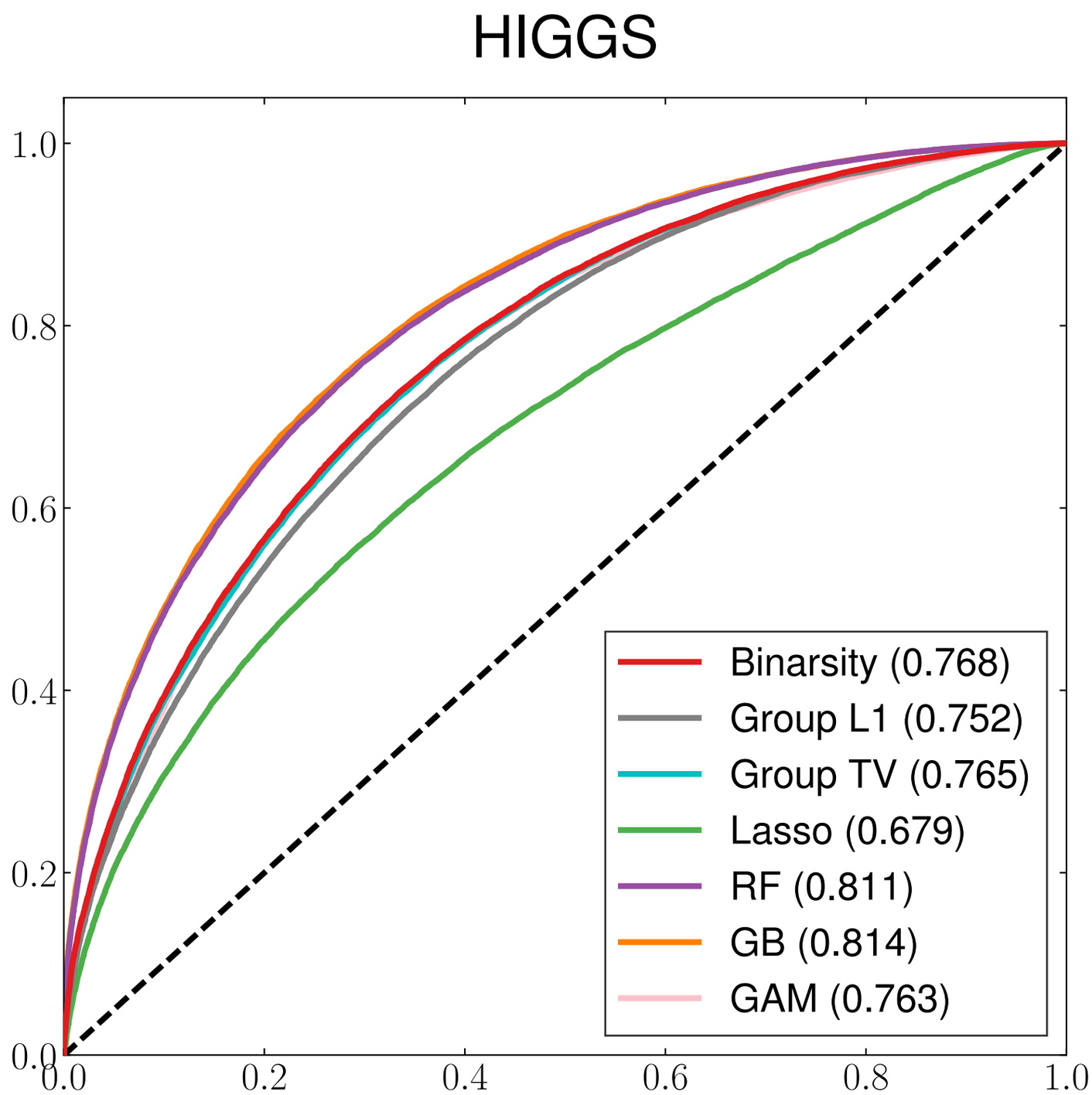
Results


SUSY
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Binarsity (0.855)
Group L1 (0.847)
Group TV (0.854)
Lasso (0.843)
RF (0.858)
GB (0.861)
GAM (0.856)

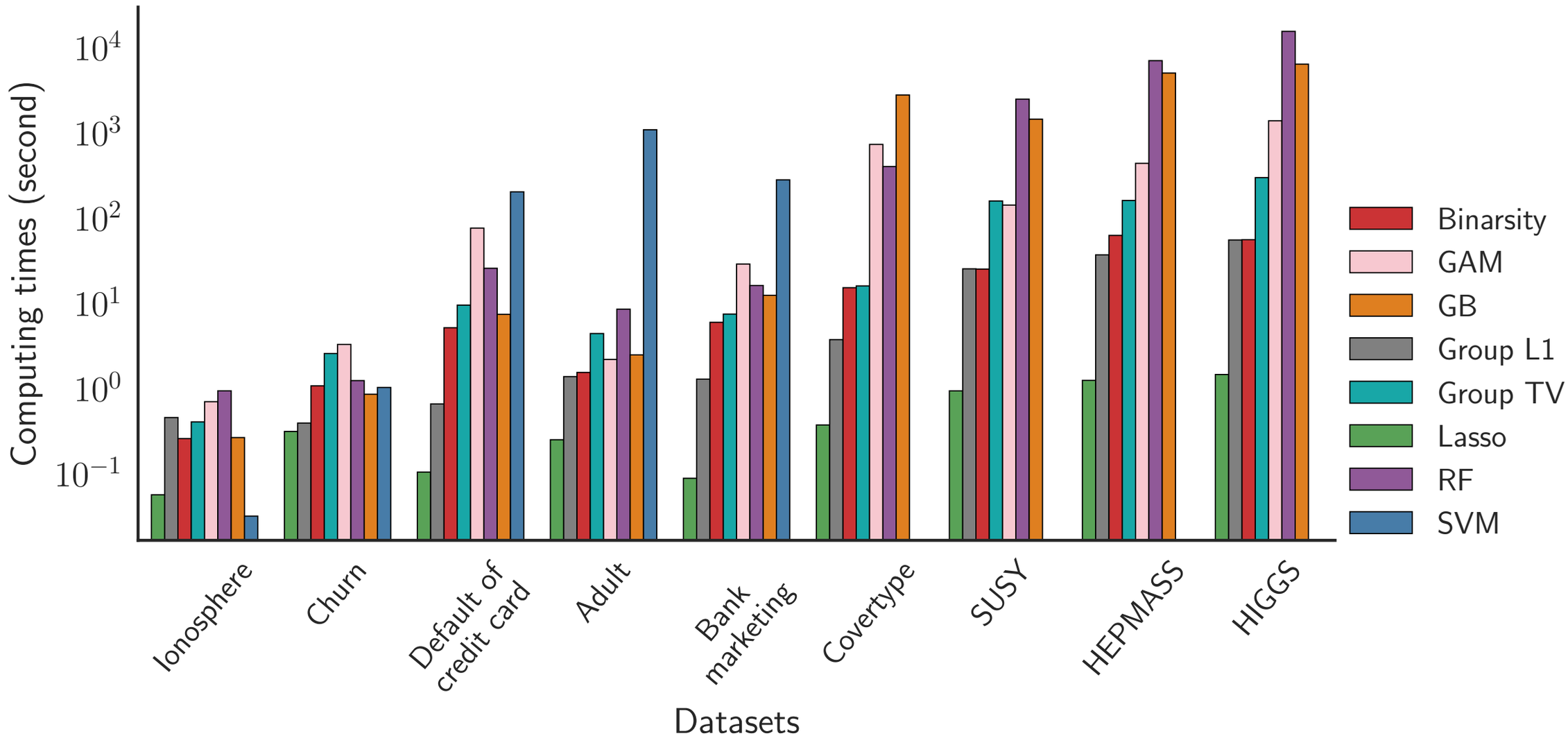
Results


HEPMASS
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Binarsity (0.963)
Group L1 (0.961)
Group TV (0.962)
Lasso (0.959)
RF (0.965)
GB (0.967)
GAM (0.963)

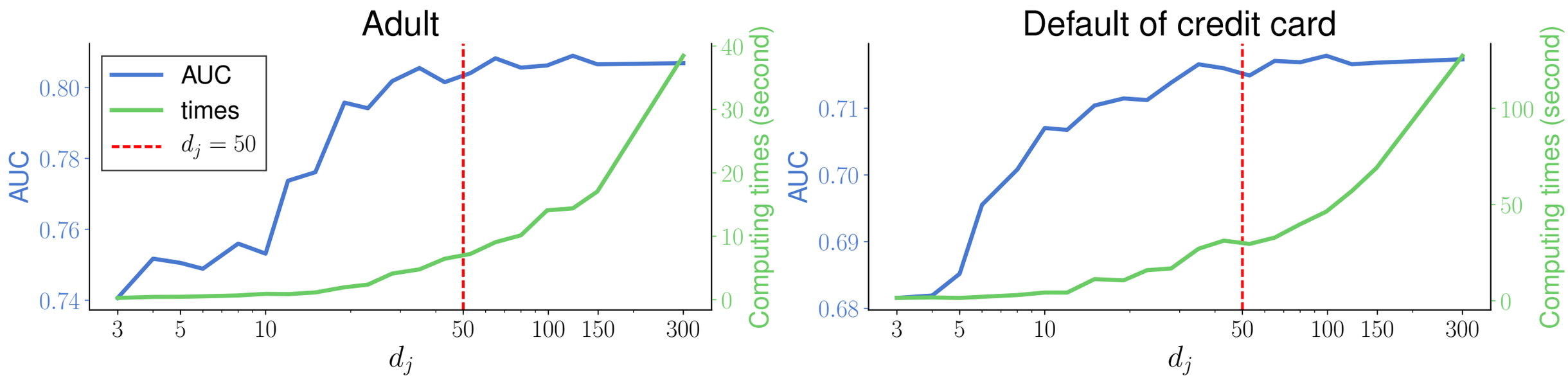
Results


HIGGS
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Binarsity (0.768)
Group L1 (0.752)
Group TV (0.765)
Lasso (0.679)
RF (0.811)
GB (0.814)
GAM (0.763)

Computing Time Comparisons



Log-scaled computing time comparisons between the methods on the considered datasets. Binarsity is between 2 and 5 times slower than Lasso but more than 100 times faster than RF and GB on larges datasets like HIGGS.

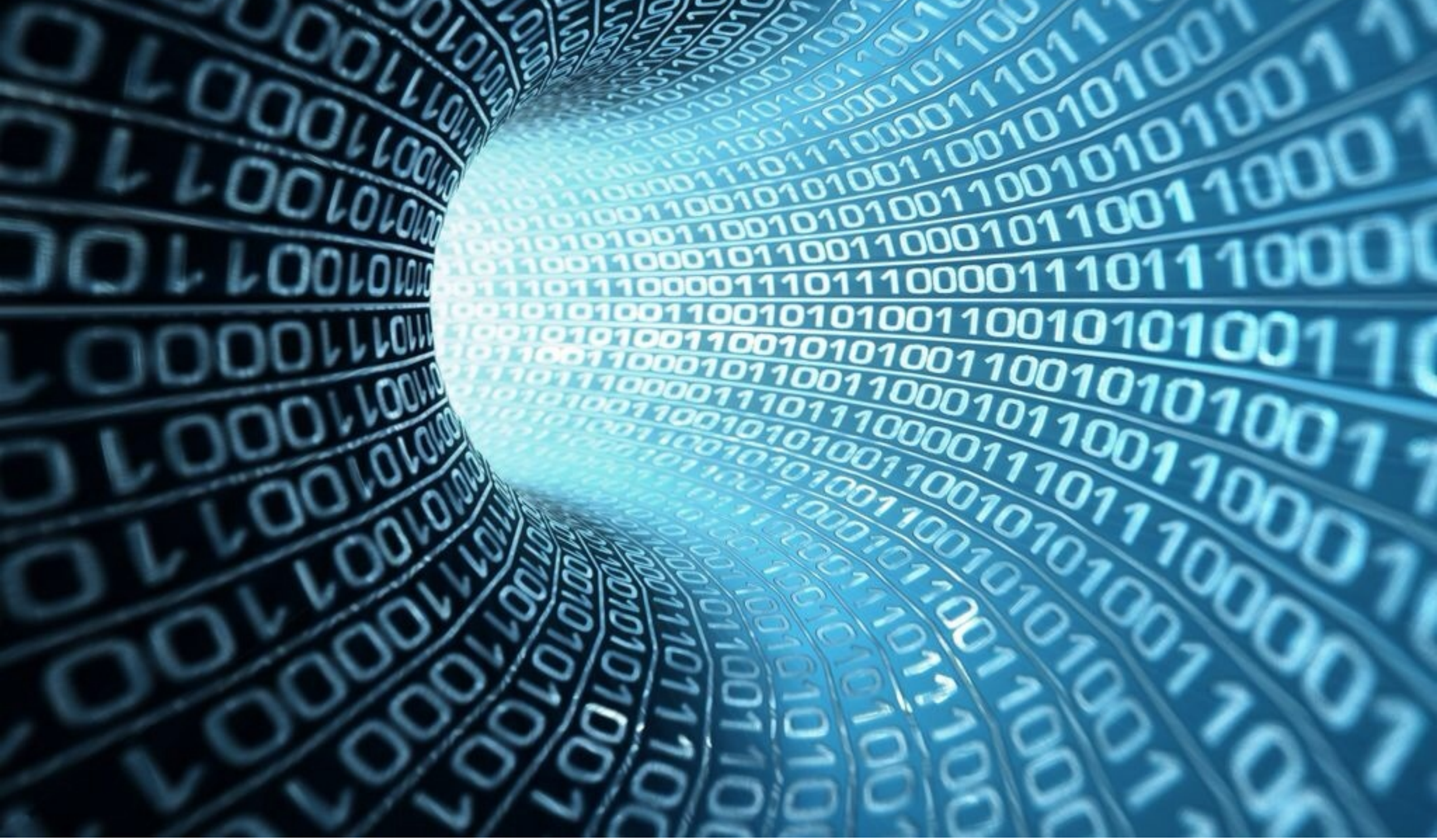

Adult
AUC
times
$d_j = 50$
AUC
0.80
0.78
0.76
0.74
Computing times (second)
40
30
20
10
0
3
5
10
50
100
150
300
d_j
Default of credit card
AUC
0.71
0.70
0.69
0.68
Computing times (second)
100
50
0
3
5
10
50
100
150
300
d_j

Take Home Message

- We introduced the binarsity penalization for one-hot encodings of continuous features

- We illustrated the good statistical properties of binarsity for generalized linear models by proving non-asymptotic oracle inequalities.

- We conducted extensive comparisons of binarsity with state-of-the-art algorithms for binary classification on several standard datasets.



Thank you!