Feature-Weighted Elastic Net for Competing Risk Outcomes

Lukas Burk

Context

The elastic net objective function is given as:

$$J(\beta_0, \beta) = \frac{1}{2}||\mathbf{y} - \beta_0\mathbf{1} - \mathbf{X}\beta||_2^2 + \lambda \sum_{j=1}^{p} \left(\alpha|\beta_j| + \frac{1-\alpha}{2}\beta_j^2 \right)$$

- $\lambda \in \mathbb{R}$ controls overall sparsity

(Zou and Hastie 2005)

Context

The elastic net objective function is given as:

$$J(\beta_0, \beta) = \frac{1}{2}||\mathbf{y} - \beta_0\mathbf{1} - \mathbf{X}\beta||_2^2 + \lambda \sum_{j=1}^{p} \left(\alpha|\beta_j| + \frac{1-\alpha}{2}\beta_j^2 \right)$$

- $\lambda \in \mathbb{R}$ controls overall sparsity
- Higher $\Rightarrow$ larger penalty on all β_j, *equally*

(Zou and Hastie 2005)

Context

The elastic net objective function is given as:

$$J(\beta_0, \beta) = \frac{1}{2}||\mathbf{y} - \beta_0 \mathbf{1} - \mathbf{X}\beta||_2^2 + \lambda \sum_{j=1}^{p} \left(\alpha|\beta_j| + \frac{1-\alpha}{2}\beta_j^2 \right)$$

- $\lambda \in \mathbb{R}$ controls overall sparsity
- Higher $\Rightarrow$ larger penalty on all β_j, *equally*

- $\alpha \in [0, 1]$ is the mixing parameter

(Zou and Hastie 2005)

Context

The elastic net objective function is given as:

$$J(\beta_0, \beta) = \frac{1}{2}||\mathbf{y} - \beta_0\mathbf{1} - \mathbf{X}\beta||_2^2 + \lambda \sum_{j=1}^{p} \left(\alpha|\beta_j| + \frac{1-\alpha}{2}\beta_j^2 \right)$$

- $\lambda \in \mathbb{R}$ controls overall sparsity
- Higher $\Rightarrow$ larger penalty on all β_j, *equally*
- $\alpha \in [0, 1]$ is the mixing parameter
- $1 \Rightarrow$ only ℓ^1 penalty (LASSO)

(Zou and Hastie 2005)

Context

The elastic net objective function is given as:

$$J(\beta_0, \beta) = \frac{1}{2}||\mathbf{y} - \beta_0\mathbf{1} - \mathbf{X}\beta||_2^2 + \lambda \sum_{j=1}^{p} \left(\alpha|\beta_j| + \frac{1-\alpha}{2}\beta_j^2 \right)$$

- $\lambda \in \mathbb{R}$ controls overall sparsity
- Higher $\Rightarrow$ larger penalty on all β_j, *equally*

- $\alpha \in [0, 1]$ is the mixing parameter
- $1 \Rightarrow$ only ℓ^1 penalty (LASSO)
- $0 \Rightarrow$ only ℓ_2^2 penalty

(Zou and Hastie 2005)

Feature-Weighted Elastic Net (`fwelnet`)

- Introduced by Tay et al. (2020)

Feature-Weighted Elastic Net (`fwelnet`)

- Introduced by Tay et al. (2020)
- Multiple applications: External information & feature-grouping

Feature-Weighted Elastic Net (`fwelnet`)

- Introduced by Tay et al. (2020)
- Multiple applications: External information & feature-grouping
- Sometimes it's desirable to adjust penalization weights on individual or groups of coefficients

Feature-Weighted Elastic Net (`fwelnet`)

- Introduced by Tay et al. (2020)
- Multiple applications: External information & feature-grouping
- Sometimes it's desirable to adjust penalization weights on individual or groups of coefficients
- Assign groups via matrix $\mathbf{Z} \in \mathbb{R}^{p \times K}$

Feature Weighting: Groups

$$\mathbf{Z} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

- Features 1, 2   group 1

Feature Weighting: Groups

$$\mathbf{Z} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

- Features 1, 2 group 1
- Features 3, 4 group 2

Feature Weighting: Groups

$$\mathbf{Z} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

- Features 1, 2 group 1
- Features 3, 4 group 2

Feature Weighting: Groups

$$\mathbf{Z} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

- Features 1, 2 group 1
- Features 3, 4 group 2

Now imagine this, but with $p >> 1000$ and e.g. genetic data.

Feature-Weighting

$$J(\beta_0, \beta) = \frac{1}{2}||\mathbf{y} - \beta_0 \mathbf{1} - \mathbf{X}\beta||_2^2 + \lambda \sum_{j=1}^{p} w_j(\theta) \left(\alpha|\beta_j| + \frac{1-\alpha}{2}\beta_j^2 \right)$$

where

Feature-Weighting

$$J(\beta_0, \beta) = \frac{1}{2}||\mathbf{y} - \beta_0\mathbf{1} - \mathbf{X}\beta||_2^2 + \lambda \sum_{j=1}^{p} w_j(\theta) \left(\alpha|\beta_j| + \frac{1-\alpha}{2}\beta_j^2\right)$$

where

$$w_j(\theta) = \frac{\sum_{l=1}^{p} \exp(\mathbf{z}_l^T \theta)}{p \exp(\mathbf{z}_j^T \theta)}$$

Defines the **penalization weight** of coefficient j based on its corresponding value in $\mathbf{Z}$ and additionally hyper-parameter θ

Penalization Weights

- $w_j(\theta)$ is chosen heuristically, suggested by the authors for desirable properties

Penalization Weights

- $w_j(\theta)$ is chosen heuristically, suggested by the authors for desirable properties
- $\mathbf{z}_j^T\theta$ functions as a *score*

Penalization Weights

- $w_j(\theta)$ is chosen heuristically, suggested by the authors for desirable properties
- $\mathbf{z}_j^T\theta$ functions as a *score*
  - $= 0$, reduces to original elastic net

Penalization Weights

- $w_j(\theta)$ is chosen heuristically, suggested by the authors for desirable properties
- $\mathbf{z}_j^T\theta$ functions as a *score*
  - $= 0$, reduces to original elastic net
  - Higher score –> lower penalization weight, feature is "more important"

Penalization Weights

- $w_j(\theta)$ is chosen heuristically, suggested by the authors for desirable properties
- $\mathbf{z}_j^T\theta$ functions as a *score*
  - $= 0$, reduces to original elastic net
  - Higher score –> lower penalization weight, feature is "more important"

Penalization Weights

- $w_j(\theta)$ is chosen heuristically, suggested by the authors for desirable properties
- $\mathbf{z}_j^T\theta$ functions as a *score*
  - $= 0$, reduces to original elastic net
  - Higher score –> lower penalization weight, feature is "more important"
- In the "feature grouping" setting, this just allows group-specific penalization weights.

Penalization Weights

- $w_j(\theta)$ is chosen heuristically, suggested by the authors for desirable properties
- $\mathbf{z}_j^T\theta$ functions as a *score*
  - $= 0$, reduces to original elastic net
  - Higher score –$>$ lower penalization weight, feature is "more important"
- In the "feature grouping" setting, this just allows group-specific penalization weights.
- Related to the "group lasso" (Jacob, Obozinski, and Vert 2009)

Feature-Weighting: Single Group

- $\mathbf{Z}$ can also be $p \times 1$: No groups, just weights

Feature-Weighting: Single Group

- $\mathbf{Z}$ can also be $p \times 1$: No groups, just weights
- Example from paper: $\mathbf{Z}$ set to noisy version of true $|\beta|$ in simulation study

Feature-Weighting: Single Group

- $\mathbf{Z}$ can also be $p \times 1$: No groups, just weights
- Example from paper: $\mathbf{Z}$ set to noisy version of true $|\beta|$ in simulation study

Feature-Weighting: Single Group

- $\mathbf{Z}$ can also be $p \times 1$: No groups, just weights
- Example from paper: $\mathbf{Z}$ set to noisy version of true $|\beta|$ in simulation study

- Higher $|\beta_j| \Rightarrow$ lower penalization for $\hat{\beta}_j$

Feature-Weighting: Single Group

- $\mathbf{Z}$ can also be $p \times 1$: No groups, just weights
- Example from paper: $\mathbf{Z}$ set to noisy version of true $|\beta|$ in simulation study

- Higher $|\beta_j| \Rightarrow$ lower penalization for $\hat{\beta}_j$
- $|\beta_j| \approx 0 \Rightarrow$ higher penalization for $\hat{\beta}_j$

Application for Multi-Task Learning

- Authors suggest multi-task learning algorithm, outline:

Application for Multi-Task Learning

- Authors suggest multi-task learning algorithm, outline:

1. $\beta_1^{(0)}, \beta_2^{(0)}$: `glmnet` solution for $(\mathbf{X}, \mathbf{y}_1), (\mathbf{X}, \mathbf{y}_2)$ respectively

Application for Multi-Task Learning

- Authors suggest multi-task learning algorithm, outline:

1. $\beta_1^{(0)}, \beta_2^{(0)}$: `glmnet` solution for $(\mathbf{X}, \mathbf{y}_1), (\mathbf{X}, \mathbf{y}_2)$ respectively
2. For $k = 0, 1, ...$:

Application for Multi-Task Learning

- Authors suggest multi-task learning algorithm, outline:

1. $\beta_1^{(0)}, \beta_2^{(0)}$: `glmnet` solution for $(\mathbf{X}, \mathbf{y}_1), (\mathbf{X}, \mathbf{y}_2)$ respectively
2. For $k = 0, 1, ...$:
   a) $\mathbf{Z}_2 = \left|\beta_1^{(k)}\right|$. Fit `fwelnet` with $(\mathbf{X}, \mathbf{y}_2, \mathbf{Z}_2)$

Application for Multi-Task Learning

- Authors suggest multi-task learning algorithm, outline:

1. $\beta_1^{(0)}, \beta_2^{(0)}$: `glmnet` solution for $(\mathbf{X}, \mathbf{y}_1), (\mathbf{X}, \mathbf{y}_2)$ respectively
2. For $k = 0, 1, ...$:
   a) $\mathbf{Z}_2 = \left|\beta_1^{(k)}\right|$. Fit `fwelnet` with $(\mathbf{X}, \mathbf{y}_2, \mathbf{Z}_2)$
      - Set $\left|\beta_2^{(k+1)}\right|$ to solution with optimal `lambda`

Application for Multi-Task Learning

- Authors suggest multi-task learning algorithm, outline:

1. $\beta_1^{(0)}, \beta_2^{(0)}$: `glmnet` solution for $(\mathbf{X}, \mathbf{y}_1), (\mathbf{X}, \mathbf{y}_2)$ respectively
2. For $k = 0, 1, ...$:
   a) $\mathbf{Z}_2 = \left|\beta_1^{(k)}\right|$. Fit `fwelnet` with $(\mathbf{X}, \mathbf{y}_2, \mathbf{Z}_2)$
      - Set $\left|\beta_2^{(k+1)}\right|$ to solution with optimal `lambda`

   b) $\mathbf{Z}_1 = \left|\beta_2^{(k+1)}\right|$. Fit `fwelnet` with $(\mathbf{X}, \mathbf{y}_1, \mathbf{Z}_1)$

Application for Multi-Task Learning

- Authors suggest multi-task learning algorithm, outline:

1. $\beta_1^{(0)}, \beta_2^{(0)}$: `glmnet` solution for $(\mathbf{X}, \mathbf{y}_1), (\mathbf{X}, \mathbf{y}_2)$ respectively
2. For $k = 0, 1, \ldots$:
   a) $\mathbf{Z}_2 = \left|\beta_1^{(k)}\right|$. Fit `fwelnet` with $(\mathbf{X}, \mathbf{y}_2, \mathbf{Z}_2)$
      - Set $\left|\beta_2^{(k+1)}\right|$ to solution with optimal `lambda`

   b) $\mathbf{Z}_1 = \left|\beta_2^{(k+1)}\right|$. Fit `fwelnet` with $(\mathbf{X}, \mathbf{y}_1, \mathbf{Z}_1)$
      - Set $\left|\beta_1^{(k+1)}\right|$ to solution with optimal `lambda`

Transfer to Competing Risks

- Setting with two causes: $\mathbf{y}_1, \mathbf{y}_2 \in \{0, 1\}$

Transfer to Competing Risks

- Setting with two causes: $\mathbf{y}_1, \mathbf{y}_2 \in \{0, 1\}$
- Assumption: Shared information for both causes:

Transfer to Competing Risks

- Setting with two causes: $\mathbf{y}_1, \mathbf{y}_2 \in \{0, 1\}$
- Assumption: Shared information for both causes:
  - If $\mathbf{x}_j$ is important for $\mathbf{y}_1$, also for $\mathbf{y}_2$?

Transfer to Competing Risks

- Setting with two causes: $\mathbf{y}_1, \mathbf{y}_2 \in \{0, 1\}$
- Assumption: Shared information for both causes:
  - If $\mathbf{x}_j$ is important for $\mathbf{y}_1$, also for $\mathbf{y}_2$?
  - $\Rightarrow$ Avoid shrinking it to 0 in cause-specific models

Transfer to Competing Risks

- Setting with two causes: $\mathbf{y}_1, \mathbf{y}_2 \in \{0, 1\}$
- Assumption: Shared information for both causes:
  - If $\mathbf{x}_j$ is important for $\mathbf{y}_1$, also for $\mathbf{y}_2$?
  - $\Rightarrow$ Avoid shrinking it to 0 in cause-specific models
- Basic idea: Adapt previous algorithm to Cox regression

Transfer to Competing Risks

- Setting with two causes: $\mathbf{y}_1, \mathbf{y}_2 \in \{0, 1\}$
- Assumption: Shared information for both causes:
  - If $\mathbf{x}_j$ is important for $\mathbf{y}_1$, also for $\mathbf{y}_2$?
  - $\Rightarrow$ Avoid shrinking it to 0 in cause-specific models
- Basic idea: Adapt previous algorithm to Cox regression
- Multi-task $\Rightarrow$ "Multi-cause"

What We Have

- 1: Adapt `fwelnet` for Coxnet/`Surv` endpoint via `glmnet`

What We Have

- 1: Adapt `fwelnet` for Coxnet/`Surv` endpoint via `glmnet`
- 2: Implement algorithm for Cox/CR setting

What We Have

- 1: Adapt `fwelnet` for Coxnet/`Surv` endpoint via `glmnet`
- 2: Implement algorithm for Cox/CR setting
- 3: Apply algorithm to some “easy” simulated data settings:

What We Have

- 1: Adapt `fwelnet` for Coxnet/`Surv` endpoint via `glmnet`
- 2: Implement algorithm for Cox/CR setting
- 3: Apply algorithm to some “easy” simulated data settings:
  - Large effects ($\beta_j = 1$ or 0.25 for “small effect”)

What We Have

- 1: Adapt `fwelnet` for Coxnet/`Surv` endpoint via `glmnet`
- 2: Implement algorithm for Cox/CR setting
- 3: Apply algorithm to some “easy” simulated data settings:
  - Large effects ($\beta_j = 1$ or 0.25 for “small effect”)
  - $N = 1000$

What We Have

- 1: Adapt `fwelnet` for Coxnet/`Surv` endpoint via `glmnet`
- 2: Implement algorithm for Cox/CR setting
- 3: Apply algorithm to some “easy” simulated data settings:
  - Large effects ($\beta_j = 1$ or 0.25 for “small effect”)
  - $N = 1000$
  - 11-14 noise variables

What We Have

- 1: Adapt `fwelnet` for Coxnet/`Surv` endpoint via `glmnet`
- 2: Implement algorithm for Cox/CR setting
- 3: Apply algorithm to some "easy" simulated data settings:
  - Large effects ($\beta_j = 1$ or 0.25 for "small effect")
  - $N = 1000$
  - 11-14 noise variables

What We Have

- 1: Adapt `fwelnet` for Coxnet/`Surv` endpoint via `glmnet`
- 2: Implement algorithm for Cox/CR setting
- 3: Apply algorithm to some “easy” simulated data settings:
  - Large effects ($\beta_j = 1$ or 0.25 for “small effect”)
  - $N = 1000$
  - 11-14 noise variables

First goal: See if we find some improvement over cause-specific `glmnet`

What Comes Next

- Wider set of simulation cases

What Comes Next

- Wider set of simulation cases
- Evaluate predictive performance (non-trivial in CR/censored setting)

References

Jacob, Laurent, Guillaume Obozinski, and Jean-Philippe Vert. 2009. "Group Lasso with Overlap and Graph Lasso." In *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*, 1–8. ACM Press. https://doi.org/10.1145/1553374.1553431.

Tay, J. Kenneth, Nima Aghaeepour, Trevor Hastie, and Robert Tibshirani. 2020. "Feature-Weighted Elastic Net: Using 'Features of Features' for Better Prediction." *arXiv:2006.01395 [Cs, Stat]*, June. http://arxiv.org/abs/2006.01395.

Zou, Hui, and Trevor Hastie. 2005. "Regularization and Variable Selection via the Elastic Net." *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 67 (2): 301–20. https://www.jstor.org/stable/3647580.