Lecture 10: Regularised regression (cont'd)

Rebecka Jörnsten, Mathematical Sciences

MSA220/MVE441 Statistical Learning for Big Data

28th April, 2022

CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG

Regularisation in classification

Recall: Regularised Discriminant Analysis (RDA)

Given training samples $(i_l, \mathbf{x}_l)$, quadratic DA models

$$p(\mathbf{x}|i) = N(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \quad \text{and} \quad p(i) = \pi_i$$

Estimates $\widehat{\boldsymbol{\mu}}_i$, $\widehat{\boldsymbol{\Sigma}}_i$ and $\widehat{\pi}_i$ are straight-forward to find,...

Recall: Regularised Discriminant Analysis (RDA)

Given training samples $(i_l, \mathbf{x}_l)$, quadratic DA models

$$p(\mathbf{x}|i) = N(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \quad \text{and} \quad p(i) = \pi_i$$

Estimates $\widehat{\boldsymbol{\mu}}_i$, $\widehat{\boldsymbol{\Sigma}}_i$ and $\widehat{\pi}_i$ are straight-forward to find,…

…but evaluating the normal density requires inversion of $\widehat{\boldsymbol{\Sigma}}_i$. If it is (near-)singular, this can lead to **numerical instability**.

Recall: Regularised Discriminant Analysis (RDA)

Given training samples $(i_l, \mathbf{x}_l)$, quadratic DA models

$$p(\mathbf{x}|i) = N(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \quad \text{and} \quad p(i) = \pi_i$$

Estimates $\widehat{\boldsymbol{\mu}}_i$, $\widehat{\boldsymbol{\Sigma}}_i$ and $\widehat{\pi}_i$ are straight-forward to find,...

...but evaluating the normal density requires inversion of $\widehat{\boldsymbol{\Sigma}}_i$. If it is (near-)singular, this can lead to **numerical instability**.

Regularisation can help here:

- Use $\widehat{\boldsymbol{\Sigma}}_i = \widehat{\boldsymbol{\Sigma}}_i^{\mathrm{QDA}} + \lambda \widehat{\boldsymbol{\Sigma}}^{\mathrm{LDA}}$ for $\lambda > 0$

Recall: Regularised Discriminant Analysis (RDA)

Given training samples $(i_l, \mathbf{x}_l)$, quadratic DA models

$$p(\mathbf{x}|i) = N(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \quad \text{and} \quad p(i) = \pi_i$$

Estimates $\widehat{\boldsymbol{\mu}}_i$, $\widehat{\boldsymbol{\Sigma}}_i$ and $\widehat{\pi}_i$ are straight-forward to find,...

...but evaluating the normal density requires inversion of $\widehat{\boldsymbol{\Sigma}}_i$. If it is (near-)singular, this can lead to **numerical instability**.

Regularisation can help here:

- Use $\widehat{\boldsymbol{\Sigma}}_i = \widehat{\boldsymbol{\Sigma}}_i^{\mathrm{QDA}} + \lambda \widehat{\boldsymbol{\Sigma}}^{\mathrm{LDA}}$ for $\lambda > 0$
- Use LDA (i.e. $\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}$) and $\widehat{\boldsymbol{\Sigma}} = \widehat{\boldsymbol{\Sigma}}^{\mathrm{LDA}} + \lambda \boldsymbol{\Delta}$ for $\lambda > 0$ and a diagonal matrix $\boldsymbol{\Delta}$

Recall: Naive Bayes LDA

Naive Bayes LDA means that we assume that $\widehat{\boldsymbol{\Sigma}} = \widehat{\boldsymbol{\Delta}}$ for a diagonal matrix $\widehat{\boldsymbol{\Delta}}$. The diagonal elements are estimated as

$$\widehat{\Delta}^{(j,j)} = \frac{1}{n-K} \sum_{i=1}^{K} \sum_{i_l = i} (\mathbf{x}_l^{(j)} - \widehat{\boldsymbol{\mu}}_i^{(j)})^2$$

which is the **pooled within-class variance**.

Recall: Naive Bayes LDA

Naive Bayes LDA means that we assume that $\widehat{\boldsymbol{\Sigma}} = \widehat{\boldsymbol{\Delta}}$ for a diagonal matrix $\widehat{\boldsymbol{\Delta}}$. The diagonal elements are estimated as

$$\widehat{\Delta}^{(j,j)} = \frac{1}{n-K} \sum_{i=1}^{K} \sum_{i_l = i} (\mathbf{x}_l^{(j)} - \widehat{\boldsymbol{\mu}}_i^{(j)})^2$$

which is the **pooled within-class variance**.

Classification is performed by predicting the class with the maximal **discriminant function** value

$$\begin{aligned} \delta_i(\mathbf{x}) &= -\frac{1}{2}(\mathbf{x} - \widehat{\boldsymbol{\mu}}_i)^\top \widehat{\boldsymbol{\Delta}}^{-1}(\mathbf{x} - \widehat{\boldsymbol{\mu}}_i) + \log(\widehat{\pi}_i) \\ &= -\frac{1}{2}\left\|\widehat{\boldsymbol{\Delta}}^{-1/2}(\mathbf{x} - \widehat{\boldsymbol{\mu}}_i)\right\|_2^2 + \log(\widehat{\pi}_i) \end{aligned}$$

where $\left(\widehat{\boldsymbol{\Delta}}^{-1/2}\right)^{(i,i)} = 1/\sqrt{\widehat{\boldsymbol{\Delta}}^{(i,i)}}$.

Shrunken centroids (I)

In high-dimensional problems ($p > n$), centroids will

- contain noise
- be hard to interpret when all variables are active

Shrunken centroids (I)

In high-dimensional problems ($p > n$), centroids will

- contain noise
- be hard to interpret when all variables are active

As in regression, we would like to perform **variable selection** and **reduce noise**.

Shrunken centroids (I)

In high-dimensional problems ($p > n$), centroids will

- contain noise
- be hard to interpret when all variables are active

As in regression, we would like to perform **variable selection** and **reduce noise**.

Recall: The class centroids solve

$$\widehat{\boldsymbol{\mu}}_i = \frac{1}{n_i} \sum_{i_l = i} \mathbf{x}_l = \arg\min_{\mathbf{v}} \frac{1}{2} \sum_{i_l = i} ||\mathbf{x}_l - \mathbf{v}||_2^2$$

Shrunken centroids (I)

In high-dimensional problems ($p > n$), centroids will

- contain noise
- be hard to interpret when all variables are active

As in regression, we would like to perform **variable selection** and **reduce noise**.

Recall: The class centroids solve

$$\widehat{\boldsymbol{\mu}}_i = \frac{1}{n_i} \sum_{i_l = i} \mathbf{x}_l = \arg\min_{\mathbf{v}} \frac{1}{2} \sum_{i_l = i} ||\mathbf{x}_l - \mathbf{v}||_2^2$$

Idea: Can we perform variable selection through ℓ_1-/lasso-style regularisation? How can we account for varying variance in features and stabilise against noise?

Shrunken centroids (II)

Nearest shrunken centroids performs variable selection and stabilises centroid estimates by solving

$$\bar{\boldsymbol{\mu}}_i = \arg\min_{\mathbf{v}} \frac{1}{2} \sum_{i_l = i} \|(\widehat{\boldsymbol{\Delta}} + s_0 \mathbf{I}_p)^{-1/2}(\mathbf{x}_l - \mathbf{v})\|_2^2 + \lambda n_i m_i \|\mathbf{v} - \widehat{\boldsymbol{\mu}}_T\|_1$$

where $s_0 = \mathrm{median}(\widehat{\boldsymbol{\Delta}}^{(1,1)}, \dots, \widehat{\boldsymbol{\Delta}}^{(p,p)})$, $m_i = \sqrt{\frac{1}{n_i} - \frac{1}{n}}$ and $\widehat{\boldsymbol{\mu}}_T = \frac{1}{n} \sum_l \mathbf{x}_l$.

Shrunken centroids (II)

Nearest shrunken centroids performs variable selection and stabilises centroid estimates by solving

$$\bar{\boldsymbol{\mu}}_i = \arg\min_{\mathbf{v}} \frac{1}{2} \sum_{i_l = i} \|(\widehat{\boldsymbol{\Delta}} + s_0 \mathbf{I}_p)^{-1/2}(\mathbf{x}_l - \mathbf{v})\|_2^2 + \lambda n_i m_i \|\mathbf{v} - \widehat{\boldsymbol{\mu}}_T\|_1$$

where $s_0 = \mathrm{median}(\widehat{\boldsymbol{\Delta}}^{(1,1)}, \dots, \widehat{\boldsymbol{\Delta}}^{(p,p)})$, $m_i = \sqrt{\frac{1}{n_i} - \frac{1}{n}}$ and $\widehat{\boldsymbol{\mu}}_T = \frac{1}{n} \sum_l \mathbf{x}_l$.

- Penalises distance of class centroid to the overall centroid $\boldsymbol{\mu}_T$

Shrunken centroids (II)

Nearest shrunken centroids performs variable selection and stabilises centroid estimates by solving

$$\bar{\boldsymbol{\mu}}_i = \arg\min_{\mathbf{v}} \frac{1}{2} \sum_{i_l=i} \|(\widehat{\boldsymbol{\Delta}} + s_0 \mathbf{I}_p)^{-1/2}(\mathbf{x}_l - \mathbf{v})\|_2^2 + \lambda n_i m_i \|\mathbf{v} - \widehat{\boldsymbol{\mu}}_T\|_1$$

where $s_0 = \mathrm{median}(\widehat{\boldsymbol{\Delta}}^{(1,1)}, \ldots, \widehat{\boldsymbol{\Delta}}^{(p,p)})$, $m_i = \sqrt{\frac{1}{n_i} - \frac{1}{n}}$ and $\widehat{\boldsymbol{\mu}}_T = \frac{1}{n} \sum_l \mathbf{x}_l$.

- Penalises distance of class centroid to the overall centroid $\boldsymbol{\mu}_T$
- $\widehat{\boldsymbol{\Delta}} + s_0 \mathbf{I}_p$ is the diagonal regularised within-class covariance matrix. Features that are highly variable across samples are scaled down (**interpretability**)

Shrunken centroids (II)

Nearest shrunken centroids performs variable selection and stabilises centroid estimates by solving

$$\bar{\boldsymbol{\mu}}_i = \arg\min_{\mathbf{v}} \frac{1}{2} \sum_{i_l = i} \|(\widehat{\boldsymbol{\Delta}} + s_0 \mathbf{I}_p)^{-1/2}(\mathbf{x}_l - \mathbf{v})\|_2^2 + \lambda n_i m_i \|\mathbf{v} - \widehat{\boldsymbol{\mu}}_T\|_1$$

where $s_0 = \mathrm{median}(\widehat{\boldsymbol{\Delta}}^{(1,1)}, \dots, \widehat{\boldsymbol{\Delta}}^{(p,p)})$, $m_i = \sqrt{\frac{1}{n_i} + \frac{1}{n}}$ and $\widehat{\boldsymbol{\mu}}_T = \frac{1}{n} \sum_l \mathbf{x}_l$.

- Penalises distance of class centroid to the overall centroid $\boldsymbol{\mu}_T$
- $\widehat{\boldsymbol{\Delta}} + s_0 \mathbf{I}_p$ is the diagonal regularised within-class covariance matrix. Features that are highly variable across samples are scaled down (**interpretability**)
- $n_i m_i$ scales λ in case of unequal class sizes

Shrunken centroids (III)

The solution for component j can be derived as

$$\overline{\boldsymbol{\mu}}_i^{(j)} = \widehat{\boldsymbol{\mu}}_T^{(j)} + m_i(\widehat{\boldsymbol{\Delta}}^{(j,j)} + s_0)\,\mathrm{ST}\left(\mathbf{t}_i^{(j)}, \lambda\right) \quad \text{where} \quad \mathbf{t}_i^{(j)} = \frac{\widehat{\boldsymbol{\mu}}_i^{(j)} - \widehat{\boldsymbol{\mu}}_T^{(j)}}{m_i(\widehat{\boldsymbol{\Delta}}^{(j,j)} + s_0)}.$$

Shrunken centroids (III)

The solution for component j can be derived as

$$\overline{\boldsymbol{\mu}}_i^{(j)} = \widehat{\boldsymbol{\mu}}_T^{(j)} + m_i(\widehat{\boldsymbol{\Delta}}^{(j,j)} + s_0)\,\mathrm{ST}\left(\mathbf{t}_i^{(j)}, \lambda\right) \quad \text{where} \quad \mathbf{t}_i^{(j)} = \frac{\widehat{\boldsymbol{\mu}}_i^{(j)} - \widehat{\boldsymbol{\mu}}_T^{(j)}}{m_i(\widehat{\boldsymbol{\Delta}}^{(j,j)} + s_0)}.$$

Note: λ is a tuning parameter and has to be determined through e.g. cross-validation.

Shrunken centroids (III)

The solution for component j can be derived as

$$\overline{\boldsymbol{\mu}}_i^{(j)} = \widehat{\boldsymbol{\mu}}_T^{(j)} + m_i(\widehat{\boldsymbol{\Delta}}^{(j,j)} + s_0)\,\mathrm{ST}\left(\mathbf{t}_i^{(j)}, \lambda\right) \quad \text{where} \quad \mathbf{t}_i^{(j)} = \frac{\widehat{\boldsymbol{\mu}}_i^{(j)} - \widehat{\boldsymbol{\mu}}_T^{(j)}}{m_i(\widehat{\boldsymbol{\Delta}}^{(j,j)} + s_0)}.$$

Note: λ is a tuning parameter and has to be determined through e.g. cross-validation.

- Typically, misclassification rate improves first with increasing λ and declines for too high values

Shrunken centroids (III)

The solution for component j can be derived as

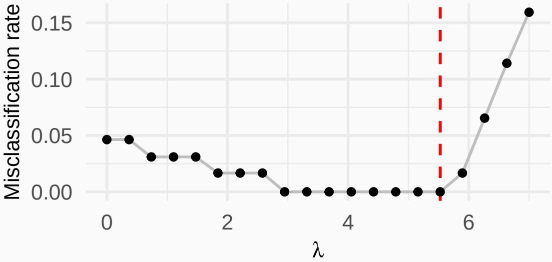
$$\overline{\boldsymbol{\mu}}_i^{(j)} = \widehat{\boldsymbol{\mu}}_T^{(j)} + m_i(\widehat{\boldsymbol{\Delta}}^{(j,j)} + s_0)\,\mathrm{ST}\left(\mathbf{t}_i^{(j)}, \lambda\right) \quad \text{where} \quad \mathbf{t}_i^{(j)} = \frac{\widehat{\boldsymbol{\mu}}_i^{(j)} - \widehat{\boldsymbol{\mu}}_T^{(j)}}{m_i(\widehat{\boldsymbol{\Delta}}^{(j,j)} + s_0)}.$$

Note: λ is a tuning parameter and has to be determined through e.g. cross-validation.

- Typically, misclassification rate improves first with increasing λ and declines for too high values
- The larger λ the more components will be equal to the respective component of the overall centroid.

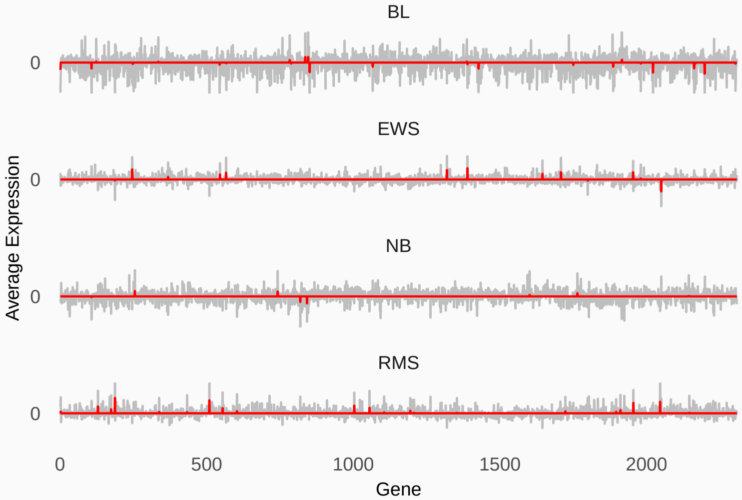
Application of nearest shrunken centroids (I)

A gene expression data set with $n = 63$ and $p = 2308$. There are four classes (cancer subtypes) with $n_{\mathrm{BL}} = 8$, $n_{\mathrm{EWS}} = 23$, $n_{\mathrm{NB}} = 12$, and $n_{\mathrm{RMS}} = 20$.



5-fold cross-validation curve and largest λ that leads to minimal misclassification rate

Application of nearest shrunken centroids (II)



Grey lines show the original centroids and red lines show the shrunken centroids

Extensions of the lasso

The lasso and groups of highly correlated variables

- The lasso does not handle groups of highly correlated variables well.

The lasso and groups of highly correlated variables

- The lasso does not handle groups of highly correlated variables well.
- **Example:** Two groups of highly correlated variables, e.g.

$$\mathbf{X} \sim N(\mathbf{0}, \boldsymbol{\Sigma}) \quad \text{where} \quad \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_1 \end{pmatrix} \in \mathbb{R}^{200\times 200},$$

where

$$\boldsymbol{\Sigma}_1 \in \mathbb{R}^{100\times 100}, \quad \boldsymbol{\Sigma}_1^{(i,i)} = 1.04 \quad \text{and} \quad \boldsymbol{\Sigma}_1^{(i,j)} = 1, \quad i \neq j.$$

The response is generated for $n = 100$ samples as

$$\mathbf{y} = \mathbf{x}_1 - \mathbf{x}_{102} + \boldsymbol{\varepsilon} \quad \text{where} \quad \boldsymbol{\varepsilon} \sim N(\mathbf{0}, 4\mathbf{I}_p).$$

The lasso and groups of highly correlated variables

- The lasso does not handle groups of highly correlated variables well.
- **Example:** Two groups of highly correlated variables, e.g.

$$\mathbf{X} \sim N(\mathbf{0}, \mathbf{\Sigma}) \quad \text{where} \quad \mathbf{\Sigma} = \begin{pmatrix} \mathbf{\Sigma}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_1 \end{pmatrix} \in \mathbb{R}^{200\times 200},$$

  where

$$\mathbf{\Sigma}_1 \in \mathbb{R}^{100\times 100}, \quad \mathbf{\Sigma}_1^{(i,i)} = 1.04 \quad \text{and} \quad \mathbf{\Sigma}_1^{(i,j)} = 1, \quad i \neq j.$$

  The response is generated for $n = 100$ samples as

$$\mathbf{y} = \mathbf{x}_1 - \mathbf{x}_{102} + \boldsymbol{\varepsilon} \quad \text{where} \quad \boldsymbol{\varepsilon} \sim N(\mathbf{0}, 4\mathbf{I}_p).$$

- **Expectation:** Since the predictors in each group are strongly correlated, all could be considered equally as predictors.

The lasso and groups of highly correlated variables

- The lasso does not handle groups of highly correlated variables well.
- **Example:** Two groups of highly correlated variables, e.g.

$$\mathbf{X} \sim N(\mathbf{0}, \boldsymbol{\Sigma}) \quad \text{where} \quad \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_1 \end{pmatrix} \in \mathbb{R}^{200\times 200},$$

  where

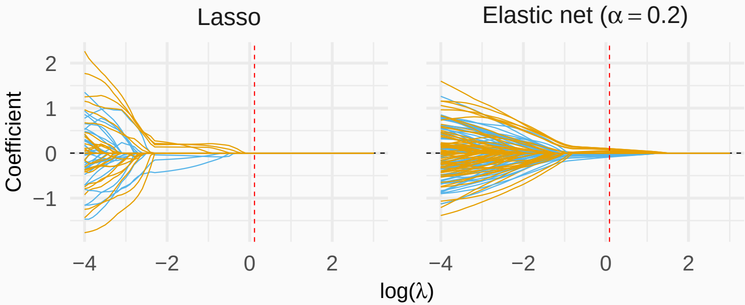
$$\boldsymbol{\Sigma}_1 \in \mathbb{R}^{100\times 100}, \quad \boldsymbol{\Sigma}_1^{(i,i)} = 1.04 \quad \text{and} \quad \boldsymbol{\Sigma}_1^{(i,j)} = 1, \quad i \neq j.$$

  The response is generated for $n = 100$ samples as

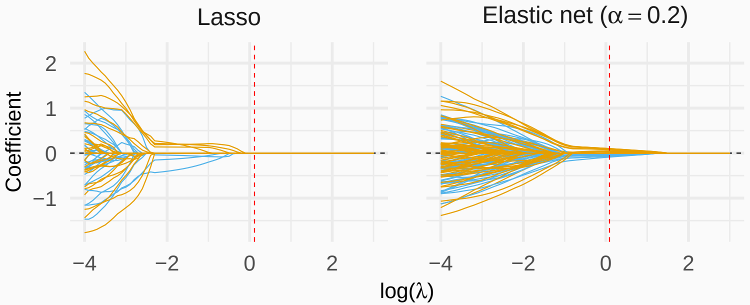
$$\mathbf{y} = \mathbf{x}_1 - \mathbf{x}_{102} + \boldsymbol{\varepsilon} \quad \text{where} \quad \boldsymbol{\varepsilon} \sim N(\mathbf{0}, 4\mathbf{I}_p).$$

- **Expectation:** Since the predictors in each group are strongly correlated, all could be considered equally as predictors.
- **Possible caveat:** The lasso makes a sparsity assumption and tries to set as many coefficients to zero as possible.

The lasso and groups of highly correlated variables in practice

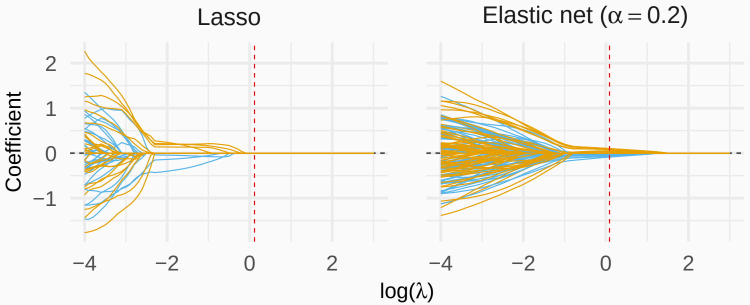


The lasso and groups of highly correlated variables in practice



- At optimal λ the lasso selects 8 non-zero coefficients 0 of which were in the true coefficient vector.
  - Very precise but 'wrong' estimates.

The lasso and groups of highly correlated variables in practice



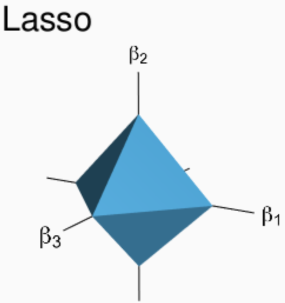
- At optimal λ the lasso selects 8 non-zero coefficients 0 of which were in the true coefficient vector.
  - Very precise but 'wrong' estimates.
- An alternative algorithm, the **elastic net** estimates 95 non-zero coefficients. (44 in the 1st group and 51 in the 2nd group, group-wise close coefficients)
  - 'Shares' responsibility among correlated variables

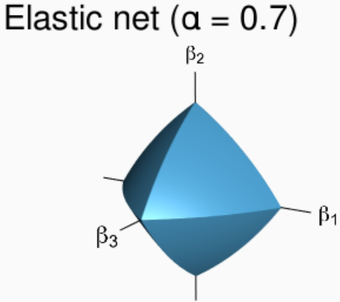
The elastic net (I)

The **elastic net** solves the problem

$$\arg\min_{\boldsymbol{\beta}} \frac{1}{2}||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2 + \lambda\left(\frac{1-\alpha}{2}||\boldsymbol{\beta}||_2^2 + \alpha||\boldsymbol{\beta}||_1\right)$$

striking a balance between lasso (**variable selection**) and ridge regression (**grouping of variables**)





Notes on the elastic net (II)

- The solution can be found through **cyclic coordinate descent**
- α is an additional tuning parameter that should be determined by cross-validation
- The lasso and ridge regression are special cases of the elastic net ($\alpha = 1$ and $\alpha = 0$, respectively).

Explicitly adding groups to the lasso

- The lasso in it's original formulation considers each variable separately

Explicitly adding groups to the lasso

- The lasso in it's original formulation considers each variable separately
- Groups in data can form through e.g.
  - Correlation
  - Categorical variables in dummy encoding
  - Domain-knowledge (e.g. genes in the same signal pathway, signals that only appear in groups in a compressed sensing problem,...)

Explicitly adding groups to the lasso

- The lasso in it's original formulation considers each variable separately
- Groups in data can form through e.g.
  - Correlation
  - Categorical variables in dummy encoding
  - Domain-knowledge (e.g. genes in the same signal pathway, signals that only appear in groups in a compressed sensing problem,...)
- Ideally the whole group is either present or not

Explicitly adding groups to the lasso

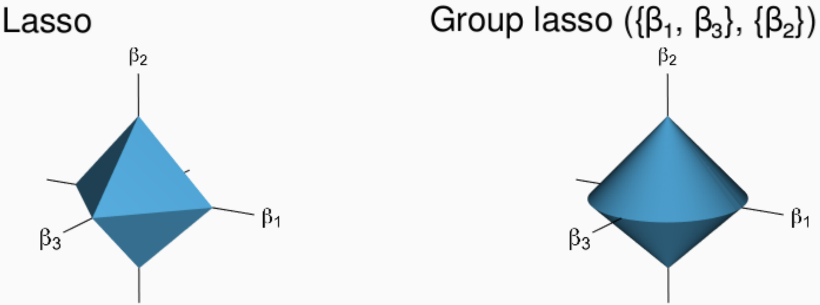
- The lasso in it's original formulation considers each variable separately
- Groups in data can form through e.g.
  - Correlation
  - Categorical variables in dummy encoding
  - Domain-knowledge (e.g. genes in the same signal pathway, signals that only appear in groups in a compressed sensing problem,...)
- Ideally the whole group is either present or not
- The elastic net can find groups, but only does so for highly correlated variables and without external influence. Sometimes more control is necessary.

The group lasso (I)

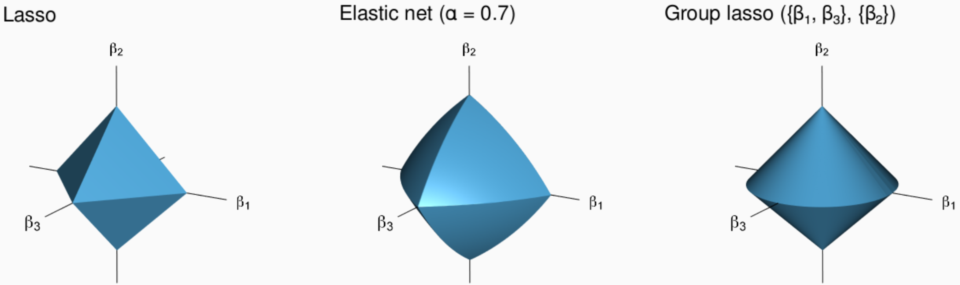
The **group lasso** solves the problem

$$\arg\min_{\boldsymbol{\beta}} \frac{1}{2}||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2 + \lambda \sum_{k=1}^{K} ||\mathbf{B}_k||_2$$

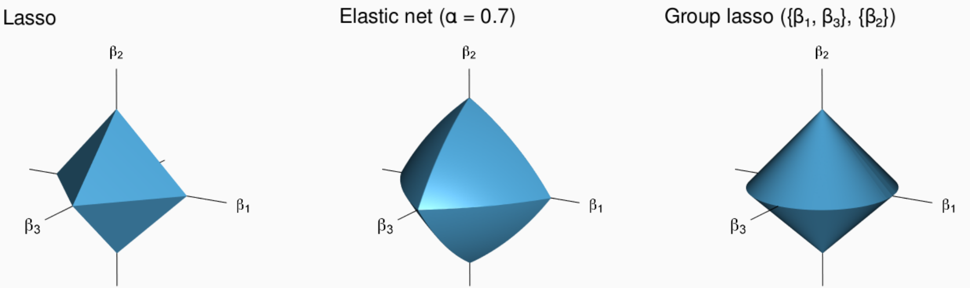
where $\mathbf{B}_k$ is a vector of coefficients β_i for the k-th group. Note that $||\beta_i||_2 = |\beta_i|$ for singleton groups.



Comparison: Lasso, elastic net and group lasso

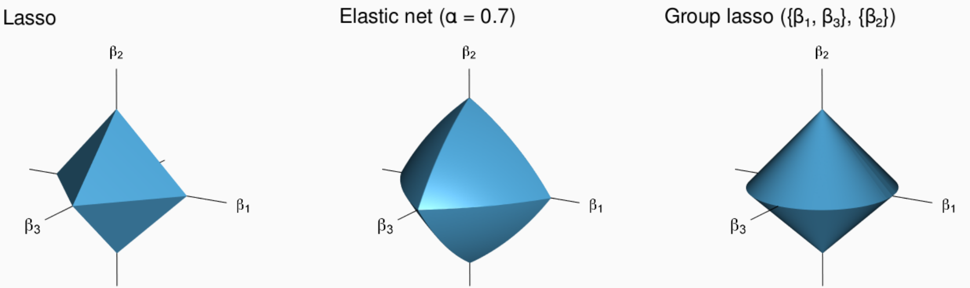


Comparison: Lasso, elastic net and group lasso



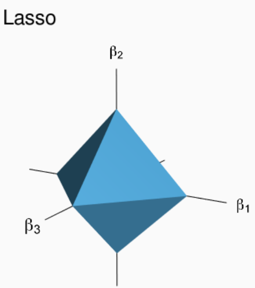
- The lasso sets variables exactly to zero either on a corner or along an edge.

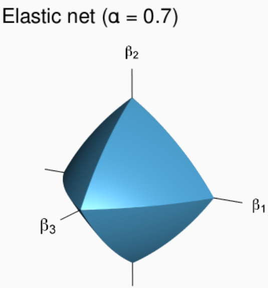
Comparison: Lasso, elastic net and group lasso

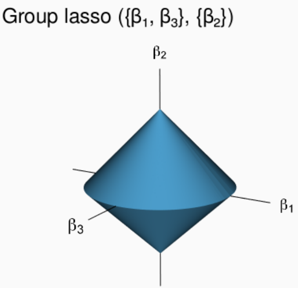


- The lasso sets variables exactly to zero either on a corner or along an edge.
- The elastic net similarly sets variables exactly to zero on a corner or along an edge. The curved edges encourage remaining coefficients to be closer together.

Comparison: Lasso, elastic net and group lasso







- The lasso sets variables exactly to zero either on a corner or along an edge.
- The elastic net similarly sets variables exactly to zero on a corner or along an edge. The curved edges encourage remaining coefficients to be closer together.
- The group lasso has actual information about groups of variables. It encourages whole groups to be zero or non-zero with similar coefficients.

Penalisation in GLMs

Penalisation can also be used in generalised linear models (GLMs), e.g. to perform **sparse logistic regression**.

Penalisation in GLMs

Penalisation can also be used in generalised linear models (GLMs), e.g. to perform **sparse logistic regression**.

Given $p(y|\boldsymbol{\beta}, \mathbf{x})$ the log-likelihood of the model is

$$\mathcal{L}(\boldsymbol{\beta}|\mathbf{y}, \mathbf{X}) = \sum_{l=1}^{n} \log(p(y_l|\boldsymbol{\beta}, \mathbf{x}_l))$$

Penalisation in GLMs

Penalisation can also be used in generalised linear models (GLMs), e.g. to perform **sparse logistic regression**.

Given $p(y|\boldsymbol{\beta}, \mathbf{x})$ the log-likelihood of the model is

$$\mathcal{L}(\boldsymbol{\beta}|\mathbf{y}, \mathbf{X}) = \sum_{l=1}^{n} \log(p(y_l|\boldsymbol{\beta}, \mathbf{x}_l))$$

Instead of penalising the minimisation of the residual sum of squares (RSS), the **minimisation of the negative log-likelihood is penalized**, i.e.

$$\arg\min_{\boldsymbol{\beta}} -\mathcal{L}(\boldsymbol{\beta}|\mathbf{y}, \mathbf{X}) + \lambda||\boldsymbol{\beta}||_1$$

Penalisation in GLMs

Penalisation can also be used in generalised linear models (GLMs), e.g. to perform **sparse logistic regression**.

Given $p(y|\boldsymbol{\beta}, \mathbf{x})$ the log-likelihood of the model is

$$\mathcal{L}(\boldsymbol{\beta}|\mathbf{y}, \mathbf{X}) = \sum_{l=1}^{n} \log(p(y_l|\boldsymbol{\beta}, \mathbf{x}_l))$$

Instead of penalising the minimisation of the residual sum of squares (RSS), the **minimisation of the negative log-likelihood is penalized**, i.e.

$$\arg\min_{\boldsymbol{\beta}} -\mathcal{L}(\boldsymbol{\beta}|\mathbf{y}, \mathbf{X}) + \lambda||\boldsymbol{\beta}||_1$$

Note: If $p(y|\boldsymbol{\beta}, \mathbf{x})$ is Gaussian and the linear model $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$ is assumed, this is equivalent to the lasso.

Sparse logistic regression

Recall: For logistic regression with $i_l \in \{0,1\}$ it holds that

$$p(1|\boldsymbol{\beta},\mathbf{x}) = \frac{\exp(\mathbf{x}^\top\boldsymbol{\beta})}{1+\exp(\mathbf{x}^\top\boldsymbol{\beta})} \quad \text{and} \quad p(0|\boldsymbol{\beta},\mathbf{x}) = \frac{1}{1+\exp(\mathbf{x}^\top\boldsymbol{\beta})}$$

Sparse logistic regression

Recall: For logistic regression with $i_l \in \{0, 1\}$ it holds that

$$p(1|\boldsymbol{\beta}, \mathbf{x}) = \frac{\exp(\mathbf{x}^\top \boldsymbol{\beta})}{1 + \exp(\mathbf{x}^\top \boldsymbol{\beta})} \quad \text{and} \quad p(0|\boldsymbol{\beta}, \mathbf{x}) = \frac{1}{1 + \exp(\mathbf{x}^\top \boldsymbol{\beta})}$$

and the penalised minimisation problem becomes

$$\arg\min_{\boldsymbol{\beta}} - \sum_{l=1}^{n} \left(i_l \mathbf{x}_l^\top \boldsymbol{\beta} - \log\left(1 + \exp(\mathbf{x}^\top \boldsymbol{\beta})\right)\right) + \lambda ||\boldsymbol{\beta}||_1$$

Sparse logistic regression

Recall: For logistic regression with $i_l \in \{0, 1\}$ it holds that

$$p(1|\boldsymbol{\beta}, \mathbf{x}) = \frac{\exp(\mathbf{x}^\top \boldsymbol{\beta})}{1 + \exp(\mathbf{x}^\top \boldsymbol{\beta})} \quad \text{and} \quad p(0|\boldsymbol{\beta}, \mathbf{x}) = \frac{1}{1 + \exp(\mathbf{x}^\top \boldsymbol{\beta})}$$

and the penalised minimisation problem becomes

$$\arg\min_{\boldsymbol{\beta}} -\sum_{l=1}^{n} \left(i_l \mathbf{x}_l^\top \boldsymbol{\beta} - \log\left(1 + \exp(\mathbf{x}^\top \boldsymbol{\beta})\right)\right) + \lambda ||\boldsymbol{\beta}||_1$$

- The minimisation problem is still convex, but non-linear in $\boldsymbol{\beta}$. Iterative quadratic approximations combined with coordinate descent can be used to solve this problem.

Sparse logistic regression

Recall: For logistic regression with $i_l \in \{0, 1\}$ it holds that

$$p(1|\boldsymbol{\beta}, \mathbf{x}) = \frac{\exp(\mathbf{x}^\top \boldsymbol{\beta})}{1 + \exp(\mathbf{x}^\top \boldsymbol{\beta})} \quad \text{and} \quad p(0|\boldsymbol{\beta}, \mathbf{x}) = \frac{1}{1 + \exp(\mathbf{x}^\top \boldsymbol{\beta})}$$

and the penalised minimisation problem becomes

$$\arg\min_{\boldsymbol{\beta}} - \sum_{l=1}^{n} \left(i_l \mathbf{x}_l^\top \boldsymbol{\beta} - \log\left(1 + \exp(\mathbf{x}^\top \boldsymbol{\beta})\right)\right) + \lambda ||\boldsymbol{\beta}||_1$$

- The minimisation problem is still convex, but non-linear in $\boldsymbol{\beta}$. Iterative quadratic approximations combined with coordinate descent can be used to solve this problem.
- Another way to perform sparse classification (like e.g. nearest shrunken centroids)

Sparse multi-class logistic regression

In multi-class logistic regression with $i_l \in \{1, \ldots, K\}$, there is a matrix of coefficients $\mathbf{B} \in \mathbb{R}^{p \times (K-1)}$ and it holds for $i = 1, \ldots, K-1$ that

$$p(i|\mathbf{B}, \mathbf{x}) = \frac{\exp(\mathbf{x}^\top \boldsymbol{\beta}_i)}{1 + \sum_{j=1}^{K-1} \exp(\mathbf{x}^\top \boldsymbol{\beta}_j)} \quad \text{and} \quad p(K|\mathbf{B}, \mathbf{x}) = \frac{1}{1 + \sum_{j=1}^{K-1} \exp(\mathbf{x}^\top \boldsymbol{\beta}_j)}$$

Sparse multi-class logistic regression

In multi-class logistic regression with $i_l \in \{1, \ldots, K\}$, there is a matrix of coefficients $\mathbf{B} \in \mathbb{R}^{p\times(K-1)}$ and it holds for $i = 1, \ldots, K-1$ that

$$p(i|\mathbf{B}, \mathbf{x}) = \frac{\exp(\mathbf{x}^\top \boldsymbol{\beta}_i)}{1 + \sum_{j=1}^{K-1} \exp(\mathbf{x}^\top \boldsymbol{\beta}_j)} \quad \text{and} \quad p(K|\mathbf{B}, \mathbf{x}) = \frac{1}{1 + \sum_{j=1}^{K-1} \exp(\mathbf{x}^\top \boldsymbol{\beta}_j)}$$

- As in two-class case, the absolute value of each entry in $\mathbf{B}$ can be penalised.

Sparse multi-class logistic regression

In multi-class logistic regression with $i_l \in \{1, \dots, K\}$, there is a matrix of coefficients $\mathbf{B} \in \mathbb{R}^{p \times (K-1)}$ and it holds for $i = 1, \dots, K-1$ that

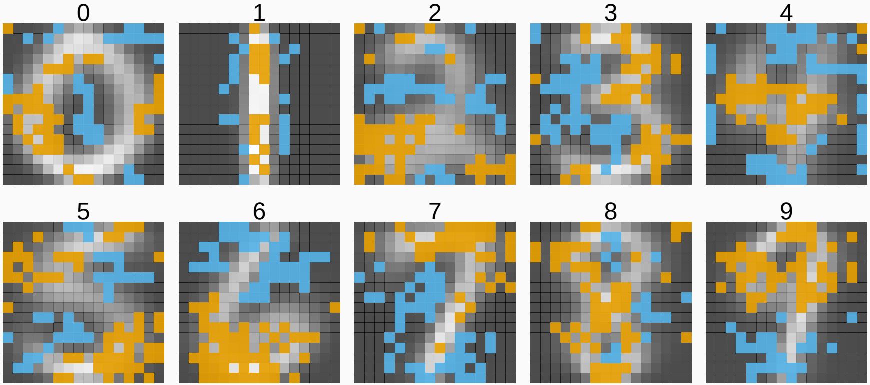
$$p(i|\mathbf{B}, \mathbf{x}) = \frac{\exp(\mathbf{x}^\top \boldsymbol{\beta}_i)}{1 + \sum_{j=1}^{K-1} \exp(\mathbf{x}^\top \boldsymbol{\beta}_j)} \quad \text{and} \quad p(K|\mathbf{B}, \mathbf{x}) = \frac{1}{1 + \sum_{j=1}^{K-1} \exp(\mathbf{x}^\top \boldsymbol{\beta}_j)}$$

- As in two-class case, the absolute value of each entry in $\mathbf{B}$ can be penalised.
- Another possibility is to use the group lasso on all coefficients for one variable, i.e. penalise with $\|\mathbf{B}_{j\cdot}\|_2$ for $j = 1, \dots, p$.

Example for sparse multi-class logistic regression

MNIST-derived zip code digits ($n = 7291$, $p = 256$)

Sparse multi-class logistic regression was applied to the whole data set and the penalisation parameter was selected by 10-fold CV.



Orange tiles show positive coefficients and blue tiles show negative coefficients. Class averages are shown in the background.

Take-home message

- Penalisation methods are not only restricted to regression, also applicable to classification
- Sparsity is a very important concept when interpretability of models is important
- Many extensions to the lasso exist, which make it more suitable for a variety of different situations