Regularization

Maarten Cruyff
1. Prediction

2. Regularization
   - ridge
   - lasso

3. Example
Prediction
Prediction

1. Train the model on data at hand
2. Predict unknown outcome on future data

Examples

- diagnosis of disease based on symptoms
- spam based on content of email
**Model**

- generalized linear model

\[ g(y) = x' \beta + \epsilon \]

**Parameter estimation**

- find \( \hat{\beta} \) that minimizes MSE / deviance
Under- and overfitting

Too few predictors in the model

- relevant predictors are missing
- parameter estimates are **biased**
- poor predictions on new data

Too many predictors in the model

- capitalization on chance, spuriousness, multicollinearity
- parameter estimates have **high variance**
- poor predictions on new data
Bias versus Variance

Figure 1: Hitting the bull’s eye.
Figure 2: Optimal prediction is compromise between bias and variance.
How to find the optimum?

Data science techniques

- stepwise procedures (AIC/BIC)
- **regularization**
- GAM’s
- trees
- boosting/bagging
- support vector machines
- deep learning
Regularization
Lasso and ridge

Regularization

- penalizing MSE/deviance with size parameter estimates

**Lasso** defined by $\ell_1$ penalty $\lambda \sum_{j=1}^p |\beta_j|$

- shrinks parameters to 0

**Ridge** defined by $\ell_2$ penalty: $\lambda \sum_{j=1}^p \beta_j^2$

- shrinks parameters towards 0

- $\lambda$ controls amount of shrinkage
- predictors are standardized
Regularization vs Stepwise

**Stepwise procedures**
- penalty on *number* of parameters (AIC/BIC)
- no hyperparameter to be estimated

**Regularization**
- penalty on *size* of parameters
- optimal shrinkage parameter to be estimated
Train/dev/test

1. Partition the data in training/test set
2. Cross validate λ's on train/validation set
3. Choose λ with smallest averaged deviance (or +1 SD)
4. Compare deviance test with competing models

Figure 3: Train/dev/test
R package glmnet

glmnet()

- fast algorithm to compute shrinkage for sequence $\lambda$
- plot parameter shrinkage as function $\lambda$

glmnet.cv()

- performs $k$-fold cross validation to determine optimal $\lambda$
- plot averaged deviance as function $\lambda$
Example
Classify email as spam/nonspm

Response variable

- 2788 mails classified as “nonspm”
- 1813 mails classified as “spam”

57 standardized frequencies of words/characters, e.g.

- !, $, ( ), #, etc.
- make, all, over, order, credit, etc.
Logistic regression model

\[
\text{logit}(\pi) = \mathbf{x}'\beta
\]

where \( \pi \) is the probability of spam.

Testing for interactions:

- 2-way: 1596 additional parameters
- 3-way: 29260 additional parameters

Restrict models to 2-way
Model comparisons

Models

- main-effects with \texttt{glm()}
- stepwise with \texttt{step()}
- ridge with \texttt{glmnet()}
- lasso with \texttt{glmnet()}
- full 2-way with \texttt{glm()}

Which model has lowest deviance on test set?
Shrinkage ridge (top) and lasso (bottom)

Results for training set (no cross validation)
Averaged deviance ridge (left) and lasso (right)

Results cross validation
## Results on test set

<table>
<thead>
<tr>
<th>Method</th>
<th>Deviance</th>
<th>Error rate</th>
<th>#pars</th>
<th>L1-norm</th>
</tr>
</thead>
<tbody>
<tr>
<td>main effects</td>
<td>269.7</td>
<td>6.3</td>
<td>58</td>
<td>104.9</td>
</tr>
<tr>
<td>ridge</td>
<td>246.7</td>
<td>7.2</td>
<td>1653</td>
<td>39.3</td>
</tr>
<tr>
<td>lasso</td>
<td>213.1</td>
<td>6.3</td>
<td>108</td>
<td>14.6</td>
</tr>
<tr>
<td>stepwise</td>
<td>572.9</td>
<td>7.7</td>
<td>129</td>
<td>3554.1</td>
</tr>
</tbody>
</table>

- **lasso**

<table>
<thead>
<tr>
<th>nonspam</th>
<th>spam</th>
</tr>
</thead>
<tbody>
<tr>
<td>nonspam</td>
<td>665</td>
</tr>
<tr>
<td>spam</td>
<td>40</td>
</tr>
</tbody>
</table>

- **main**

<table>
<thead>
<tr>
<th>nonspam</th>
<th>spam</th>
</tr>
</thead>
<tbody>
<tr>
<td>nonspam</td>
<td>666</td>
</tr>
<tr>
<td>spam</td>
<td>41</td>
</tr>
</tbody>
</table>
Conclusions

Regularization

- reduces variance without substantially increasing bias
- ability to handle large number of predictors
- fast algorithm

Extensions

- mixing $\ell_1$ and $\ell_2$ penalties (e.g. elastic net)
- grouped lasso (e.g. hierarchical models)
- similarities with Bayesian models
Thanks for your attention!