Extended inference for lasso and elastic-net regularized Cox and generalized linear models

Martin Sill, Thomas Hielscher, Natalia Becker, Manuela Zucknick

Division of Biostatistics, German Cancer Research Center, Im Neuenheimer Feld 280, 69120 Heidelberg, Germany

Abstract

We have developed an R package `c060` with the aim of improving R software functionality for high-dimensional risk prediction modelling, in particular for prognostic modelling of patient survival data using high-throughput genomic data.

Penalized regression models such as implemented in the popular CRAN package `glmnet` provide a statistically appealing method to build prediction models from high-dimensional data sources. The `glmnet` package provides an efficient state-of-the-art algorithm for fitting penalized Cox and generalized linear models. However, in a practical application the data analysis will not stop at the point, where the model has been fitted. One is for example interested in the stability of selected features or in assessing the prediction performance of a model and we provide functions to deal with both of these tasks with the possibility of speeding up computing time thorugh parallel computing. Another feature that we provide is an approach to select the optimal parameter combination for elastic net penalties using an efficient interval-search algorithm.

These functions have been useful in our daily work here at the German Cancer Research Center where prognostic modelling of patient survival data is of particular interest. Although we focus on a survival data application of penalized Cox PH regression models in this article, the functions in our R package are applicable to all types of regression models implemented in the `glmnet` package.

Keywords:

1. Introduction

Penalized regression models provide a statistically appealing method to build prediction models from high-dimensional data sources. Since the introduction of the lasso for linear regression models (Tibshirani, 1996), the methodology has been extended to generalized linear regression models, time-to-event endpoints (Tibshirani, 1997) etc. In addition to the well-known L_1- (lasso) and L_2-norms (ridge) penalty functions, various other penalties have been proposed in recent years to select features and/or estimate their effects. In particular, we will use the elastic net penalty function (Zou and Hastie, 2005), which is a linear combination of the L_1- and L_2-norms.

With ever increasing data, the properties of the algorithm to actually fit the model have become almost as important as the statistical model itself. In 2010, Friedman, Hastie and Tibshirani proposed a coordinate descent algorithm (Friedman et al., 2010) for generalized linear regression models, which has since then been extended to penalized Cox PH regression models (Simon et al., 2011). Due to its efficiency this algorithm is considered one of the state-of-the-art approaches to estimate penalized regression models with lasso, ridge or elastic net penalty terms.

This algorithm has also been implemented in R in the `glmnet` package. The package provides functions to tune and fit regression models, plot the results, and make predictions. However, in practical applications, where often an independent validation data set is lacking, some additional features and routines are desirable as part of a complete data analysis. We have assembled some functions that enhance the existing functionality of the `glmnet` package or allow to use it within the framework of other existing R packages. These functions have been useful in our daily work at the German Cancer Research Center where prognostic modelling of patient survival data is of particular interest. Therefore, for illustration purposes we focus on penalized Cox PH regression models in this article. But most of the functions are applicable to all types of regression models implemented in the `glmnet` package.

We provide R functions to perform stability selection (Meinshausen and Bühlmann, 2010) in a computationally efficient way using `glmnet` which allows to select the most stable features at a given error level. We also provide an approach to select the optimal parameter combination (α, λ) for elastic net penalties using an interval-search algorithm (Froehlich and Zell, 2005) which is often faster and more accurate than a standard grid search (Ref for a comparison study?). Another very useful addition for real-life applications

of `glmnet` for building prognostic models is the provision of wrapper functions to allow the computation of resampling-based prediction error curves with the framework of the R package `peperr` (Porzelius et al., 2009). The `peperr` package makes it computationally feasible to assess the predictive accuracy of a penalized Cox PH regression model via resampling methods even for very large-scale applications by employing parallel computing. We also provide the possibility to spead up stability selection by parallel computing using the functionalities of the R base package `parallel`.

2. Data application

Throughout this article we use the gene expression data set on cytogenetically normal acute myeloid leukemia (CN-AML) by Metzeler et al. (2008) and corresponding clinical data in order to illustrate a typical application of penalized Cox PH regression models with the aim of developing a prognostic model for patient survival while at the same time identifying the most influential gene expression features. To simulate the typical situation that only one data set is available for model training and evaluation, we only use the data set that was used as validation data in the original publication by Metzeler et al. (2008). The data can be accessed from the Gene Expression Omnibus (GEO) data repository (`http://www.ncbi.nlm.nih.gov/geo`) by the National Center for Biotechnology Information (NCBI). We find the Metzeler *et al.* data set under GEO accession number GSE12417.

The data set contains gene expression data for 79 patient samples measured with Affymetrix HG-U133 Plus 2.0 microarrays. The median survival time of these 79 patients was 17.6 months with 40% censoring.

3. Methods and algorithms

3.1. Penalized generalized linear models and Cox models

An efficient implementation for fitting generalized linear models and Cox proportional hazards models with regularization by the lasso or elastic net penalty terms is provided by the R package `glmnet` (Friedman et al., 2010; Simon et al., 2011). This implementation uses a coordinate descent algorithm for fitting the models for a specified penalty parameter value λ. The computation of an entire regularization path across a range of values $\{\lambda_1, \lambda_2 ..., \lambda_k\}$ with `glmnet` is very fast, because previously computed solutions for $\{\lambda_1, ..., \lambda_{j-1}\}$ are used as a 'hot' starting value for the computation of λ_j.

Models are fitted by maximizing the penalized log-likelihood function for generalized linear models and the penalized partial log-likelihood for Cox models. Cross-validation can be performed to decide which model (i.e. which penalty parameter values) to choose by using the cross-validated (partial) log-likelihood as the loss function.

The penalized (partial) log-likelihood is given by

$$l_n(\beta) - \sum_{j=1}^{p} p_{\lambda^*}(|\beta_j|) \tag{1}$$

where $l_n(\beta)$ denotes the (partial) log-likelihood given n observations. The dimension of the parameter vector β is p and $p_\lambda(|\cdot|)$ is the penalty function with tuning parameter λ^*. Correspondingly, the objective function employed in `glmnet` is (with $\lambda = \frac{2}{n}\lambda^*$):

$$-\frac{2}{n} l_n(\beta) + \sum_{j=1}^{p} p_\lambda(|\beta_j|). \tag{2}$$

3.2. L_2-penalized Cox regression

Penalized maximum likelihood estimation in Cox regression with the ridge penalty

$$p_\lambda(|\beta|) = \lambda\beta^2 \tag{3}$$

was introduced by Verweij and van Houwelingen (1994). The ridge penalty results in parameter estimates that are biased towards zero, but does not set values to exactly zero, and hence does not perform variable selection. On the other hand, it has been found to produce models with good prediction performance in high-dimensional genomic applications (e.g. Bøvelstad et al., 2007), in particular if predictors are highly correlated.

3.3. L_1-penalized Cox regression

Tibshirani (1997) proposed to use an L_1-penalized Cox model with

$$p_\lambda(|\beta|) = \lambda|\beta| \tag{4}$$

and described a technique, called the lasso for "least absolute shrinkage and selection operator", for parameter estimation. The L_1-penalty has the advantage over the L_2-penalty of shrinking some of the coefficients to zero, i.e. it performs automatic variable selection.

3.4. The elastic net

Zou and Hastie (2005) introduced the elastic net, which employs a combination of the L_1- and L_2-penalty

$$p_{\lambda_1,\lambda_2}(|\beta|) = \lambda_1|\beta| + \lambda_2\beta^2. \tag{5}$$

Zou and Hastie (2005) rescale the initial solutions from the optimization of the doubly-penalized log-likelihood function by the factor $1+\lambda_2$, in order to reduce the effect of the double shrinkage. Like lasso the elastic net performs automatic variable selection by setting some coefficient estimates to zero. But the additional L_2-penalty term distributes the weight to more variables, such that the elastic net tends to select more variables than the lasso. This is especially the case in situations with high correlation, where the lasso would select only one variable of a set of highly correlated variables, while the ridge penalty would give them equal weight.

Throughout this manuscript we use an alternative parametrization of the elastic net penalty function equivalently to the formulation used in the `glmnet` package:

$$p_{\alpha,\lambda}(|\beta|) = \lambda \times ((1-\alpha)\frac{1}{2}|\beta|^2 + \alpha|\beta|). \tag{6}$$

Here, $\alpha \in [0,1]$ determines the influence of the L_1 penalty relative to the L_2 penalty. Small α values will result in models with many variables, getting closer to the non-sparse ridge solution as α tends to zero.

3.4.1. The interval-search algorithm to select the optimal elastic net parameter combination

3.5. Stability selection

The stability selection proposed by Meinshausen and Bühlmann (2010) is a general approach that combines variable selection methodssuch as L_1 penalized Cox models with resampling. By applying the corresponding variable selection method to subsamples that were drawn without replacement, selection probabilities for each feature can be estimated. These selection probabilties are used to define a set of stable features. Meinshausen and Bühlmann (2010) provide a theoretical framework for controling Type I error rates of falsely assigning features to the estimated set of stable features. The selection probability of each feature along the regularization path, e.g. the range of possible penalization parameters $\Lambda = \{\lambda_1, \lambda_2 ..., \lambda_k\}$, is called stability path. Given an arbitrary threshold $\pi_{thr} \in (0.5, 1)$ and the set of

penalization parameters Λ, the set of stable features estimated with the stability selection is:

$$\hat{S}_{\beta}^{stable} = \left\{ i : \max_{\lambda_j \in \Lambda} \hat{\Pi}_i^{\lambda_j} \geq \pi_{thr} \right\}, \tag{7}$$

where $\hat{\Pi}_i^{\lambda_j}$ denotes the estimated selection probability of the ith feature at the jth λ. Then according to Theorem 1 in Meinshausen and Bühlmann (2010), the expected number of falsely selected features $E(V)$ will be bounded by:

$$E(V) \leq \frac{1}{(2\pi_{thr} - 1)} \frac{q_{\Lambda}^2}{p}, \tag{8}$$

where q_{Λ} the average of the number of non-zero coefficients w.r.t. to the drawn subsamples. Interpreting Equation 8 the expected number of falsely selected coefficients decreases by either reducing the average number of selected coefficients q_{Λ} or by increasing the threshold π_{thr}. Suppose that π_{thr} is fixed, then the stability selection controls $E(V)$ as long as the average number of selected coefficients is less then e_{Λ}. This is an upper bound

$$e_{\Lambda} = \sqrt{E(V) p (2\pi_{thr} - 1)}, \tag{9}$$

which can be controlled by reducing the length of the regularization path Λ. In multiple testing the expected number of falsely selected variables is also known as the per-family error rate (PFER) and if divided by the total number of variables p it will become the per-comparison error rate (PCER) Dudoit et al. (2003). The stability selection allows to control these Type I error rates. For instance, suppose the threshold $\pi_{thr} = 0.8$ is fixed, then choosing Λ such that $q_{\Lambda} \leq \sqrt{0.6p}$ will control $E(V) = 1$. Moreover, by choosing Λ so that $q_{\Lambda} \leq \sqrt{0.6p\alpha}$ will control the family wise error rate (FWER) at level α, $P(|V| > 0) \leq \alpha$.

3.6. Prediction error curves for survival models

The time-dependent Brier Score (Graf et al., 1999) can be used to assess and compare the prediction accuracy of prognostic model. The Brier Score at time point t is a weighted mean squared error between predicted survival probability and observed survival status. Weighting depends on the estimated censoring distribution to account for the observations under risk or more general estimates thereof (Gerds and Schumacher, 2006). Computing the error for each time point over the entire follow-up horizon yields a predicton error curve. Prediction errors based on the Kaplan-Meier estimates ignoring any additional covariate information function as reference.

The empirical time-dependent Brier score $BS(t)$ is defined as a function of time $t > 0$ by

$$BS(t) = \frac{1}{n}\sum_{i=1}^{n}\left[\frac{\hat{S}(t|x_i)^2 I(t_i \leq t \wedge \delta_i = 1)}{\hat{G}(t_i)} + \frac{(1-\hat{S}(t|x_i))^2 I(t_i > t)}{\hat{G}(t)}\right],$$

with individual survival time t_i, censoring indicator δ_i and estimated survival probability $\hat{S}(t|x_i)$ at time t based on the prognostic model given covariate values x_i. $\hat{G}(t)$ denotes the Kaplan-Meier estimate of the censoring distribution which is based on the observations $(t_i; 1-\delta_i)$, I stands for the indicator function.

In case no independent validation data are available, resampling-based prediction error curves are used to adequately assess the accuracy. The .632+ bootstrap estimator (Efron and Tibshirani, 1997), which is a weighted sum of the apparent error and the average out-of-bag bootstrap error, can be used in this context, balancing a too optimistic and a too conservative error estimation.

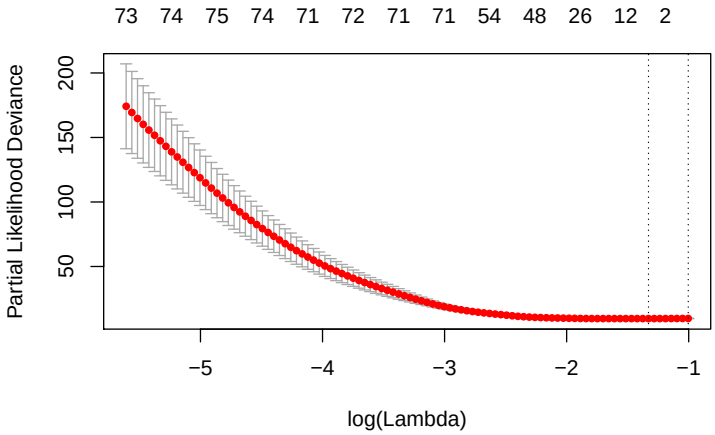

Figure 1: Cross-validated partial log-likelihood function, including upper and lower standard deviations, as a function of $\log\lambda$ for the AML data set.

4. Application and demonstration of software

4.1. Starting off: Lasso-penalised Cox model

We can apply the `glmnet` function to fit a lasso-penalized Cox model to the CN-AML data set. The function call with default penalty parameter settings will fit the lasso model for 100 λ data derived values:

```
> fit <- glmnet(y=y, x=t(exprs(eset)), family="cox")
```

We tune the lasso penalty parameter by 10-fold cross-validation using the `cv.glmnet` function. The loss function, i.e. the cross-validated partial log-likelihood, is shown in Figure 1 including upper and lower standard deviations as a function of $\log\lambda$ for the AML data set. The penalty parameter value minimizing the loss function is λ =0.265 and corresponds to a final lasso model with the following 5 selected features:

```
  203640_at 204419_x_at 222462_s_at   226169_at   233371_at
-0.11339033 -0.01664530  0.27420521  0.04300559 -0.01216429
```

The selected features are highlighted as red lines in the coefficient paths shown in Figure 2. The coefficient path shows the development of the regression coefficient estimates with increasing regularization.

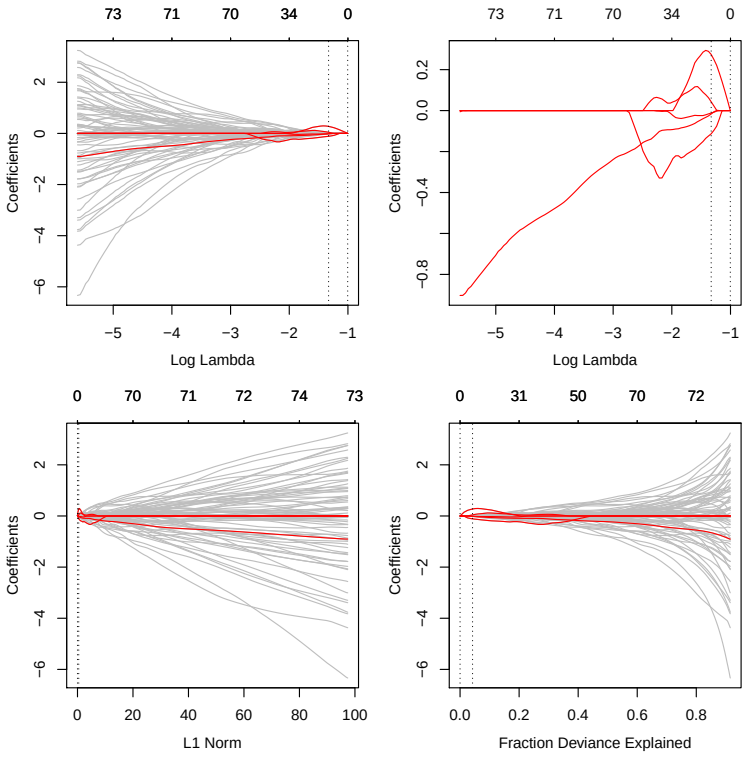

Figure 2: Coefficient paths for lasso penalised Cox PH regression model applied to the AML data set.

4.2. Resampling based prediction errors

[Manuela: Start of this section by explaining what would be done to assess the prediction performance of other types of models (LM and GLM), why survival data is special, and why we don't need a new R function for LM/GLM models.]

Once the final prognostic model is selected, we need to assess its prediction accuracy for future patients, frequently also in comparison with established clinico-pathological prognostic markers. In many applications no independent validation data set is available. The same data set need to be used to develop and assess the prognostic model. This is even more problematic for high-dimensional data, where the risk of overfitting is much more present. Resampling-based methods can be used to unbiasedly estimate the predictive accuracy of the prognostic model in this situation. This is also called internal validation or pre-validation.

The `R` package `peperr` (Porzelius et al., 2009) provides a modular framework for survival and binary endpoints, i.e. prognostic and classification models. Wrapper functions for new or customized prediction model algorithms can be defined and passed to the generic call function `peperr`. In case of prognostic models, algorithm specific wrapper functions for model fitting, tuning and prediction are required. Wrapper functions for selected machine learning approaches are already implemented.

Prediction accuracy is per default assessed with prediction error curves based on the time-dependent Brier score (Graf et al., 1999). But it is also possible to define and use customized accuracy measures.

We defined additional wrapper functions for the `glmnet` algorithm for fitting (`fit.glmnet`) and tuning (`complexity.glmnet`) the model, and predicting survival probabilities (`predictProb.glmnet`) based on the fitted model and the estimated baseline hazard from the training data.

We estimate the L_1-penalized Cox PH regression model for overall survival starting with the 10.000 most varying probe sets using `glmnet`. The .632+ bootstrap estimator is calculated based on subsampling (Binder and Schumacher, 2008) using only 100 bootstrap samples for illustration.

The `peperr` package is designed for high-dimensional covariates data and allows for various set-ups of parallel computations. Also, additional arguments can be passed directly to the `glmnet` call by specifing additional arguments for the fitting and/or tuning procedure. Here, we include patient's age as mandatory model variable into the prognostic model, i.e. age is not subject to penalization, and run the calculation on 3 CPUs in parallel using a socket cluster set-up.

```
> obj  <- peperr(response=Surv(eset$os, eset$os_status),
+          x=data.frame(eset$age,t(exprs(eset))),
+          fit.fun=fit.glmnet,args.fit=list(standardize=F,family="cox",
+          penalty.factor=rep(0:1,times=c(1,dim(eset)[1]))),
+          complexity=complexity.glmnet,
+          args.complexity=list(standardize=F,nfolds=10,
+          family="cox",penalty.factor=rep(0:1,times=c(1,dim(eset)[1]))),
+          trace=F,RNG="fixed",seed=0815,cpus=3,parallel=T,clustertype="SOCK",
+          load.list=list(functions=c("basesurv")),
+          indices=resample.indices(n=dim(eset)[2],sample.n=100,method="sub632"))
```

Individual bootstrap results can be visualized with the `plot.peperr` function from the `peperr` package showing the selected complexity parameters, out-of-bag prediction error curves as well as the prediction error integrated over time, and the predictive partial log-likelihood (PLL) values. In order to calculate the predictive PLL values again an algorithm specific wrapper (here `PLL.coxnet`) needs to be defined.

In addition, we provide a slightly modified version of the prediction error curves plot function from the `peperr` package which allows to display the number still at risk (`plot.peperr.curves`) as shown in figure 3.

Note, that for classification models, the same wrapper functions for fitting and tuning the model are called. Model performance measures shipped with the `peperr` packages are misclassification rate and Brier score.

We extended functionality of the Brier score (`aggregation.brier`) and misclassification rate (`aggregation.misclass`) calculation for the `glmnet` algorithm, and defined AUC under the ROC curve (`aggregation.auc`) as additional performance measure. For binary responses, the `peperr` package does not quite provide the same modular flexibility as for time-to-event endpoints. The predicted class probability is calculated within the performance/aggregation function by calling the algorithm specific predict function. Whenever a new algorithm is incorporated the aggregation function has to be modified and overwritten accordingly.

```
> plot.peperr.curves(obj, at.risk=T)
```

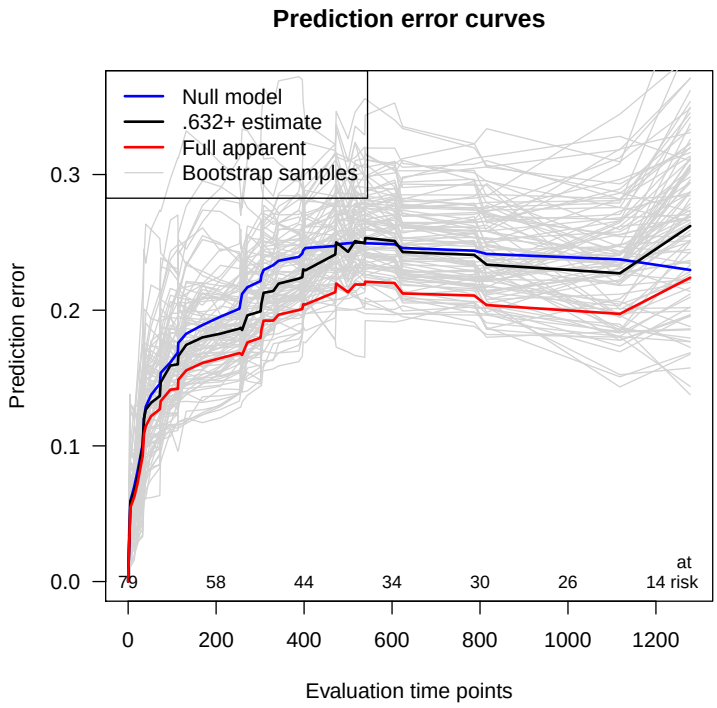

Figure 3: Prediction error curves based on time-dependent Brier score.

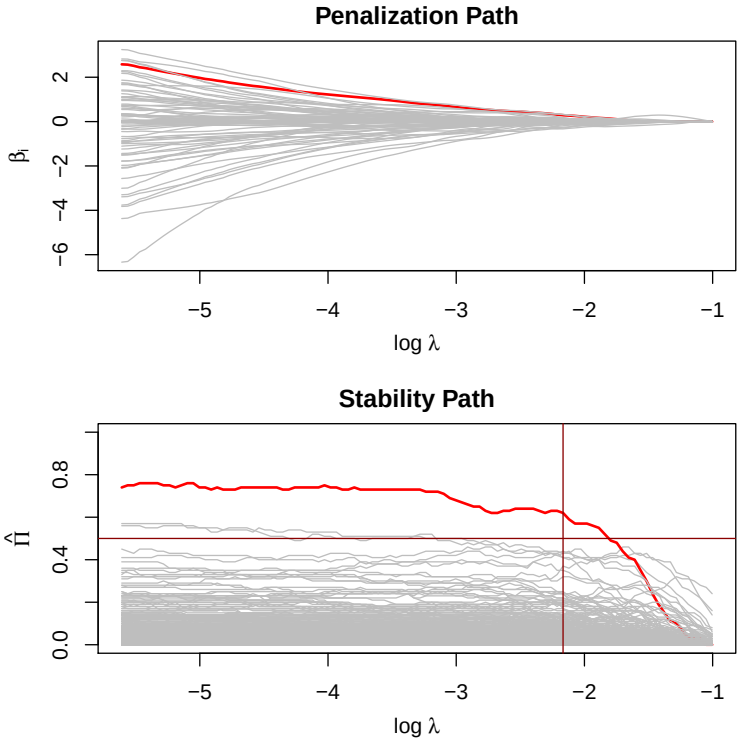

Figure 4: Coefficient and stability paths for lasso penalised Cox PH regression model applied to the AML data set.

4.3. Stability selection

Stable features (with $\hat{\Pi} > 0.5$ at λ =0.115) are:

```
206932_at
     2823
```

Table 1: Summary of visited points in the parameter space

	alpha	lambda	deviance	n.features
1	0.67605	0.5421144	9.71152	0
2	0.17194	0.5174744	9.66298	1
3	0.81895	0.4939544	9.70872	1
4	0.31188	0.4715034	9.69917	2
5	0.61671	0.4500729	9.70060	2
6	0.47035	0.4296164	9.70494	4
7	0.51860	0.4100896	9.70339	6
8	0.63074	0.3914504	9.70020	9
9	0.05830	0.3736584	9.64177	12
10	0.10655	0.3566751	9.65779	14

4.4. Elastic net penalised Cox PH regression model

4.4.1. Tune both λ and α by interval search

The task is to find such a setting of tuning parameters (α, λ), for which the 10-fold cross validation error of the model is minimal. Instead of using a fixed grid, an interval search approach is applied. REFF

The parameter space of tuning parameters is defined as follows:

```
      lower upper
alpha     0     1
```

The second tuning parameter λ will be found for each given α via regularization path. Thus, the two dimensional parameter space has the form $(0,1) \times R$. The seed is 1234. For each given α an optimal lambda is definded as largest value of lambda such that error is within 1 standard error of the minimum. (parameter type.min = 'lambda.1se').

The visited points in the parameter space (α, λ) and resulting cross-validated deviances are summarised in table 1.

The minimal mean cross-validated deviance over the folds of 9.633 is reached for optimal parameter pair $(\alpha, \lambda) = (0.013, 28.235)$.

The 'visited' points with corresponding deviance and number of selected features in each model are presented in figure 1.

```
> summary(cofn)

      Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
-0.0941500 -0.0071050 -0.0001162 -0.0002743  0.0064470  0.0913500
```

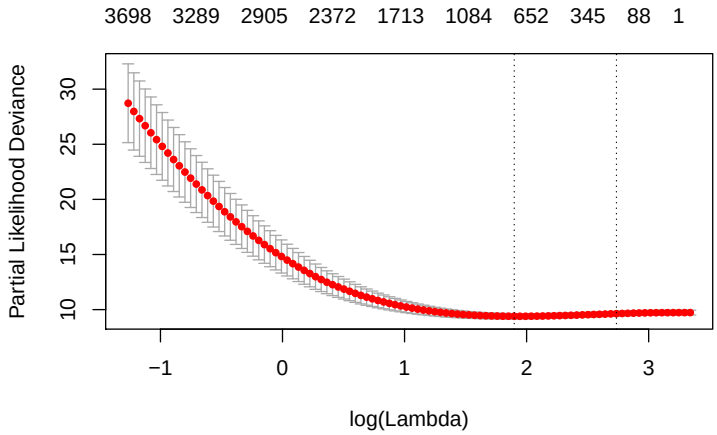

Figure 5: deviance as a function of $\log(\alpha)$ for the AML data set.

```
> head(sort(cofn))
```

```
       <NA>        <NA>        <NA>        <NA>        <NA>        <NA>
-0.09415166 -0.09328023 -0.09295101 -0.09193752 -0.09130928 -0.09004991
```

```
> tail(sort(cofn))
```

```
      <NA>       <NA>       <NA>       <NA>       <NA>       <NA>
0.08676626 0.08727578 0.08815779 0.08842483 0.09008347 0.09134615
```

A feature selected by the stability algorithm is in the set of selected features

```
> '206932_at' %in% names.cof
```

```
[1] TRUE
```

TODO : plots

5. Conclusions

References

Binder, H., Schumacher, M., 2008. Adapting prediction error estimates for biased complexity selection in high-dimensional bootstrap samples. Statistical Applications in Genetics and Molecular Biology 7.

Bøvelstad, H.M.M., Nygård, S., Størvold, H.L.L., Aldrin, M., Borgan, O., Frigessi, A., Lingjærde, O.C.C., 2007. Predicting survival from microarray data - a comparative study. Bioinformatics 23, 2080–2087.

Dudoit, S., Shaffer, J.P., Boldrick, J.C., 2003. Multiple Hypothesis Testing in Microarray Experiments. Statistical Science 18, 71–103.

Efron, B., Tibshirani, R., 1997. Improvements on cross-validation: The. 632+ bootstrap method. Journal of the American Statistical Association , 548–560.

Friedman, J., Hastie, T., Tibshirani, R., 2010. Regularization paths for generalized linear models via coordinate descent. Journal of Statistical Software 33, 1–22.

Froehlich, H., Zell, A., 2005. Efficient parameter selection for support vector machines in classification and regression via model-based global optimization, in: Proceedings of the International Joint Conference of Neural Networks, pp. 1431âĂŞ–1438.

Gerds, T., Schumacher, M., 2006. Consistent estimation of the expected brier score in general survival models with right-censored event times. Biometrical Journal 48, 1029–1040.

Graf, E., Schmoor, C., Sauerbrei, W., Schumacher, M., 1999. Assessment and comparison of prognostic classification schemes for survival data. Statistics in Medicine 18, 2529–2545.

Meinshausen, N., Bühlmann, P., 2010. Stability selection. Journal of the Royal Statistical Society - Series B: Statistical Methodology 72, 417–473.

Metzeler, K., Hummel, M., Bloomfield, C., Spiekermann, K., Braess, J., Sauerland, M.C., Heinecke, A., Radmacher, M., Marcucci, G., Whitman, S., Maharry, K., Paschka, P., Larson, R., Berdel, W., Buchner, T., Wormann, B., Mansmann, U., Hiddemann, W., Bohlander, S., Buske, C., 2008. An 86 probe set gene expression signature predicts survival in cytogenetically normal acute myeloid leukemia. Blood 112(10), 4193–4201.

Porzelius, C., Binder, H., Schumacher, M., 2009. Parallelized prediction error estimation for evaluation of high-dimensional models. Bioinformatics 25, 827–829.

Simon, N., Friedman, J., Hastie, T., Tibshirani, R., 2011. Regularization paths for cox's proportional hazards model via coordinate descent. Journal of Statistical Software 39, 1–13.

Tibshirani, R., 1996. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society, Series B: Methodological 58, 267–288.

Tibshirani, R., 1997. The lasso method for variable selection in the Cox model. Statistics in Medicine 16, 385–395.

Verweij, P.J.M., van Houwelingen, H.C., 1994. Penalized likelihood in Cox regression. Statistics in Medicine 13, 2427–2436.

Zou, H., Hastie, T., 2005. Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society, Series B 67(2), 301–320.