ggRandomForests: Survival with Random Forests

John Ehrlinger and **Eugene H. Blackstone**
Cleveland Clinic

Abstract

Random Forests (Breiman 2001) (RF) are a fully non-parametric statistical method requiring no distributional assumptions on covariate relation to the response. RF are a robust, nonlinear technique that optimizes predictive accuracy by fitting an ensemble of trees to stabilize model estimates. Random Forests for survival (Ishwaran and Kogalur 2007; Ishwaran, Kogalur, Blackstone, and Lauer 2008) (RF-S) are an extension of Breiman's RF techniques to survival settings, allowing efficient non-parametric analysis of time to event data. The **randomForestSRC** package (Ishwaran and Kogalur 2014) is a unified treatment of Breiman's random forests for survival, regression and classification problems.

Predictive accuracy make RF an attractive alternative to parametric models, though complexity and interpretability of the forest hinder wider application of the method. We introduce the **ggRandomForests** package, tools for creating and plotting data structures to visually understand random forest models grown in R with the **randomForestSRC** package. The **ggRandomForests** package is structured to extract intermediate data objects from **randomForestSRC** objects and generate figures using the **ggplot2** (Wickham 2009) graphics package.

This document is formatted as a tutorial for using the **randomForestSRC** for building random forests for survival and **ggRandomForests** package for investigating how the forest is constructed. This tutorial uses the Primary Biliary Cirrhosis (PBC) Data from the Mayo Clinic (Fleming and Harrington 1991) available in the **randomForestSRC** package. We use Variable Importance measure (VIMP) (Breiman 2001) as well as Minimal Depth (Ishwaran, Kogalur, Gorodeski, Minn, and Lauer 2010), a property derived from the construction of each tree within the forest, to assess the impact of variables on forest prediction. We will also demonstrate the use of variable dependence plots (Friedman 2000) to aid interpretation RF results in different response settings. We also will investigate interactions between covariates to demonstrate the strength of the Random Forest method in survival settings.

Keywords: random forest, survival, VIMP, minimal depth, R, **randomForestSRC**.

About this document

This document is a package vignette for the **ggRandomForests** (`http://CRAN.R-project.org/package=ggRandomForests`) package for "Visually Exploring Random Forests". **ggRandomForests** will help uncover variable associations in random forest models. The package is designed for use with the **randomForestSRC** (`http://CRAN.R-project.org/package=randomForestSRC`) package (Ishwaran and Kogalur 2014) for survival, regression and classification forests and uses the **ggplot2**(`http://CRAN.R-project.org/package=ggplot2`) package (Wickham 2009) for plotting diagnostic and variable association results. **ggRandomForests** is structured to extract data objects from **randomForestSRC** objects and provides S3 functions for printing and plotting these objects.

The vignette is a tutorial for using the **ggRandomForests** package with the **randomForest-SRC** package for building and post-processing a survival random forest. In this tutorial, we explore a random forest for survival model for the primary biliary cirrhosis (PBC) of the liver data set (Fleming and Harrington 1991), available in the **randomForestSRC** package. We grow a survival random forest and demonstrate how **ggRandomForests** can be used when determining variable associations, interactions and how the survival response depends on predictive variables within the model. The tutorial demonstrates the design and usage of many of **ggRandomForests** functions and features how to modify and customize the resulting ggplot2 graphic objects along the way.

The latest version of this vignette is available within the **ggRandomForests** package on the Compreshensive R Archive Network (CRAN) (`http://cran.r-project.org`). Once the package has been installed, the vignette can be viewed directly from within R with the following command:

```
R> vignette("survivalRandomForest", package="ggRandomForests")
```

A development version of the **ggRandomForests** package is also available on Github (`https://github.com`). We invite comments, feature requests and bug reports for this package at `https://github.com/ehrlinger/ggRandomForests`.

1. Introduction

Random Forests (Breiman 2001) (RF) are a fully non-parametric statistical method which requires no distributional assumptions on covariate relation to the response. RF is a robust, nonlinear technique that optimizes predictive accuracy by fitting an ensemble of trees to stabilize model estimates. Random Survival Forests (RSF) (Ishwaran and Kogalur 2007; Ishwaran *et al.* 2008) are an extension of Breiman's RF techniques to survival settings, allowing efficient non-parametric analysis of time to event data. The **randomForestSRC** (`http://CRAN.R-project.org/package=ggRandomForests`) package (Ishwaran and Kogalur 2014) is a unified treatment of Breiman's random forests for survival, regression and classification problems.

Predictive accuracy make RF an attractive alternative to parametric models, though complexity and interpretability of the forest hinder wider application of the method. We introduce the **ggRandomForests** (`http://CRAN.R-project.org/package=ggRandomForests`) package for visually exploring random forest models. The **ggRandomForests** package is structured to extract intermediate data objects from **randomForestSRC** objects and generate figures using the **ggplot2** (`http://CRAN.R-project.org/package=ggplot2`) graphics package (Wickham 2009).

Many of the figures created by the **ggRandomForests** package are also available directly from within the **randomForestSRC** package. However **ggRandomForests** offers the following advantages:

- Separation of data and figures: **ggRandomForests** contains functions that operate on either the `randomForestSRC::rfsrc` forest object directly, or on the output from **random-**

ForestSRC post processing functions (i.e. `plot.variable`, `var.select`, `find.interaction`) to generate intermediate **ggRandomForests** data objects. S3 functions are provide to further process these objects and plot results using the **ggplot2** graphics package. Alternatively, users can use these data objects for their own custom plotting or analysis operations.

- Each data object/figure is a single, self contained object. This allows simple modification and manipulation of the data or **ggplot2** objects to meet users specific needs and requirements.

- The use of **ggplot2** for plotting. We chose to use the **ggplot2** package for our figures to allow users flexibility in modifying the figures to their liking. Each S3 plot function returns either a single **ggplot2** object, or a `list` of **ggplot2** objects, allowing users to use additional **ggplot2** functions or themes to modify and customize the figures to their liking.

This document is formatted as a tutorial for using the **randomForestSRC** package for building and post-processing random survival forest models with the **ggRandomForests** package for investigating how the forest is constructed. In this tutorial, we will investigate the primary biliary cirrhosis (PBC) of the liver data set (Fleming and Harrington 1991), available in the **randomForestSRC** package. We present the data in Section 2 before building a random survival forest in Section 3.

Random forests are not parsimonious, but use all variables available in the construction of a response predictor. We demonstrate a random forest variable selection process (Section 4) using the Variable Importance measure (VIMP) (Breiman 2001) in Section 4.1 as well as Minimal Depth (Ishwaran *et al.* 2010) in Section 4.2. Minimal depth is a property derived from the construction of each tree within the forest, which we use to assess the impact of variables on forest prediction.

Once we have an idea of which variables the forest is using for prediction, we will use variable dependence plots (Friedman 2000) (Section 5) to understand how a variable is related to the response. Marginal variable dependence (Section 5.1) plots give us an idea of the overall trend of a variable/response relation, while partial dependence plots (Section 5.2) show us a risk adjusted relation. These figures often show strongly non-linear variable/response relations that are not easily obtained through a parametric approach. We are also interested in examining variable interactions (Section 6) within the forest model. Using a minimal depth approach, we can quantify how closely variables are related within the forest, and generate marginal dependence and partial dependence (risk adjusted) conditioning plots (coplots) (Chambers 1992; Cleveland 1993) to examine these interactions graphically (Section 7).

2. Data Summary: Primary Biliary Cirrhosis (PBC) Data

Data was obtained from a Mayo Clinic randomized trial in primary biliary cirrhosis (PBC) of the liver conducted between 1974 and 1984. A total of 424 PBC patients, referred to Mayo Clinic during that ten-year interval, met eligibility criteria for the randomized placebo controlled trial of the drug D-penicillamine. The first 312 cases in the data set participated in the randomized trial and contain largely complete data. The data is described in Fleming and

Harrington (1991) in Chapter 0.2 and Chapter 4.4. We will use the `pbc` data set transcribed from Appendix D, which is included in the **randomForestSRC** package.

```
R> data(pbc, package = "randomForestSRC")
```

For our analysis, we will do some data manipulations for formating results. Since the data contains about 12 years of follow up, we prefer using `years` instead of `days` survival. We also convert the `age` variable to years, and the `treatment` variable to a factor containing levels of `c("DPCA", "placebo")`. We outline the variable names, type and description in Table 1.

	label	type
status	event indicator (F=censor, T=death)	logical
treatment	Treament (DPCA, Placebo)	factor
age	age in years	numeric
sex	Female	logical
ascites	Asictes	logical
hepatom	Hepatomegaly	logical
spiders	Spiders	logical
edema	Edema	factor
bili	serum bilirubin (mg/dl)	numeric
chol	serum cholesterol (mg/dl)	integer
albumin	albumin (gm/dl)	numeric
copper	urine copper (ug/day)	integer
alk	alkaline phosphatase (U/liter)	numeric
sgot	SGOT (U/ml)	numeric
trig	triglicerides (mg/dl)	integer
platelet	platelets per cubic ml/1000	integer
prothrombin	prothrombin time (sec)	numeric
stage	histologic stage	factor
years	survival time (years)	numeric

Table 1: PBC Data field descriptions

2.1. Exploratory Data Analysis

It is good practice to view your data before beginning an analysis, what Tukey (1977) refers to as Exploratory Data Analysis (EDA). To facilitate this, we use **ggplot2** figures with the `facet_wrap` command to create two sets of panel plots, one for categorical variables using histograms (Figure 1), and another of scatter plots for continuous variables (Figure 2). Each variable is plotted along a selected continuous variable on the X-axis, in this case the length of follow up (survival time in `years`). These figures help to find outliers, missing values and other data anomalies within each variable before getting deep into the analysis.

In categorical EDA plots (Figure 1) we are looking for patterns of missing data. We often use surgical date for our X-axis variable to look for periods of low enrollment. The variable was not available in this data set, so we used follow up time (`years`) instead. Another good choice may have been to use the `age` variable.

In continuous EDA plots (Figure 2) we look for missingness and extreme values as in the `trig` variable. For survival, we color and shape the points corresponds to the censoring indicator (`status` variable in Figure 1), red x indicates an event, and a blue circle indicates a censored observation.

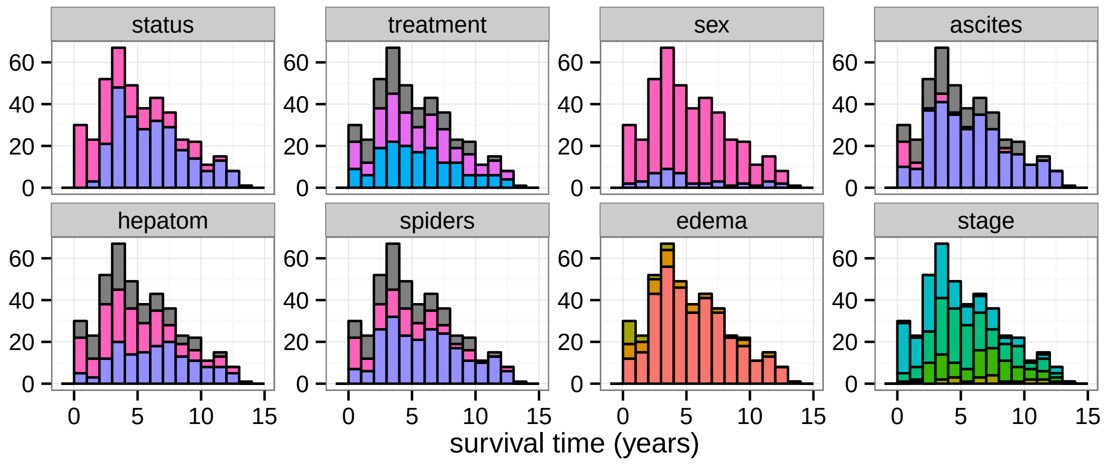

Figure 1: Categorical variable EDA plots. Bars indicate counts within 1 year of followup for each categorical variable. Bars are colored according to the class membership within each variable. Missing values are colored dark grey.

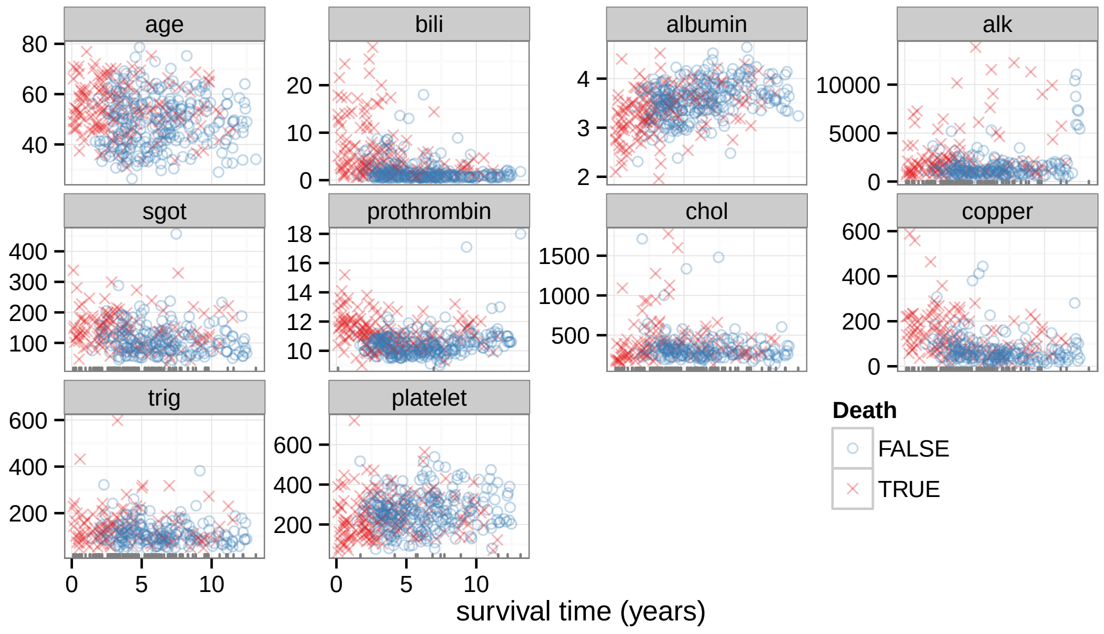

Figure 2: Continuous variable EDA plots. Points indicate variable value against the follow up time in years. Points are colored according to the death event in the `status` variable. Missing values are indicated by the rug marks along the X-axis

[redacted]	full	trial
treatment	106	0
ascites	106	0
hepatom	106	0
spiders	106	0
chol	134	28
copper	108	2
alk	106	0
sgot	106	0
trig	136	30
platelet	11	4
prothrombin	2	0
stage	6	0

Table 2: PBC missing values

There does appear to be a large amount of missing data in some variables of the `pbc` data set, indicated with dark grey bars in Figure 1, and rug marks in Figure 2. Table 2 details the number of missing values in each variable of the `pbc` data set. The `full` columns details the full data set, though there are 106 patients that were not randomized into the trial. If we remove those observations, most of the missing values are also removed. We focus on the 312 observations from the clinical trial for the remainder of this document. We will return to handling missing values in Section 3.1.

2.2. Fleming and Harrington (1991) model summary

Before turning to the random forest modeling, we conclude the data set investigation by a summary of Fleming and Harrington (1991) results from Chapter 4.4. We start by generating Kaplan–Meier (KM) survival estimates comparing the treatment groups of DPCA and placebo. We use the **ggRandomForests** `gg_survival` function to generate these estimates from the data set.

```
R> # Include only the randomized patients.
R> pbc.trial <- pbc[-which(is.na(pbc$treatment)),]
R>
R> # Create the gg_survival object
R> gg_dta <- gg_survival(interval="years",
+                        censor="status",
+                        strat="treatment",
+                        data=pbc.trial, conf.int=.95)
```

The code block first reduces the `pbc.trial` data set to only include observations from the clinical trial. The **ggRandomForests** package is designed to use a two step process in figure generation. The first step is data generation, the `gg_dta` object is a `gg_survival` data object. The `gg_survival` function uses the `data` set, the follow up `interval`, `censor` indicator and an optional grouping argument (`strat`). By default `gg_survival` also calculates 95% confidence band, which we can control with the `conf.int` argument.

In the figure generation step, we use the **ggRandomForests** S3 plot routine `plot.gg_survival` as shown in the following code block. The plot function uses the data object to plot the survival estimate curves for each group and corresponding confidence interval ribbons. We have used additional **ggplot2** commands to modify the axis and legend labels (`labs`), the legend location (`theme`) and control the plot range of the y-axis (`coord_cartesian`) for this figure. Figure 3 is analogous to Fleming and Harrington (1991) Figure 4.4.1 showing there is little difference between the treatment and control groups.

```
R> plot(gg_dta) +
+   labs(y="Survival Probability", x="Observation Time (years)",
+        color="Treatment", fill="Treatment")+
+   theme(legend.position=c(.2,.2))+
+   coord_cartesian(ylim=c(0,1.01))
```

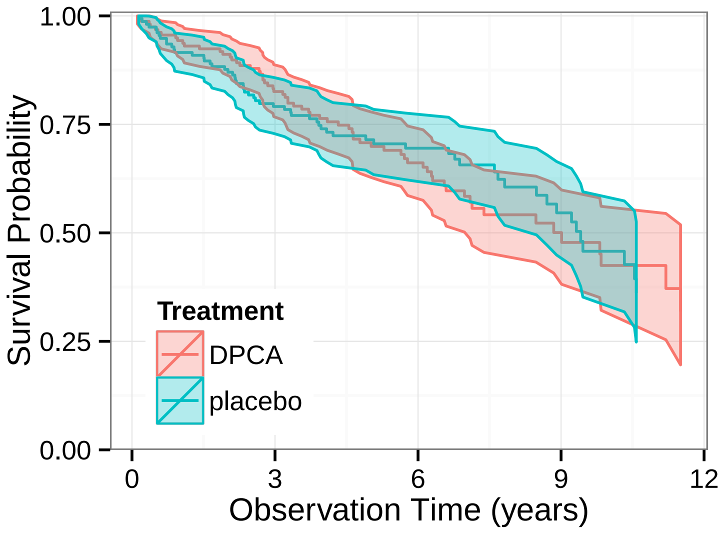

Figure 3: Kaplan–Meier pbc data survival estimates comparing the treatment with placebo. Mean survival with shaded 95% condfidence band.

In Chapter 4, Fleming and Harrington (1991) use partial likelihood methods to build a linear model with log transformations on some variables. The final, biologically reasonable model is detailed in Table 3 for later comparison with our random forest model results.

	Coef.	Std. Err.	Z stat.
Age	0.0333	0.0087	3.8400
log(Albumin)	-3.0553	0.7241	-4.2200
log(Bilirubin	0.8792	0.0987	8.9000
Edema	0.7847	0.2991	2.6200
log(Prothrombin Time)	3.0157	1.0238	2.9500

Table 3: Regression model with log transformations of continuous variables, 312 randomized cases with PBC.

In Figure 3, we demonstrated grouping on a categorical variable (`treatment`). However, with the exception of `edema`, all variables in the Fleming and Harrington (1991) model are continuous. To demonstrate plotting grouped survival on a continuous variable, we examine KM estimates of survival by stratified bilirubin grouping from Fleming and Harrington (1991) Figure 4.4.2.

The following code block duplicates the `pbc.trial` data for this exercise. We set up the `bili` groups using the `cut` function with intervals matching the reference figure. For this example we combine the data generation and plot steps into a single line of code. The `error` argument of the `plot.gg_survival` is used to control display of the confidence bands. We suppress the intervals for this figure with `error="none"` and again modify the plot display with **ggplot2** commands as before to generate Figure 4.

```
R> # Duplicate the trial data
R> pbc.bili <- pbc.trial
R>
R> # Group by bilirubin values
R> pbc.bili$bili_grp <- cut(pbc.trial$bili,
+                         breaks=c(0,.8,1.3,3.4, max(pbc.trial$bili)))
R>
R> # plot the gg_survival object directly
R> plot(gg_survival(interval="years",censor="status",
+                  strat="bili_grp", data=pbc.bili ),
+      error="none") +
+   labs(y="Survival Probability",
+        x="Observation Time (years)",
+        color="Bilirubin")
```

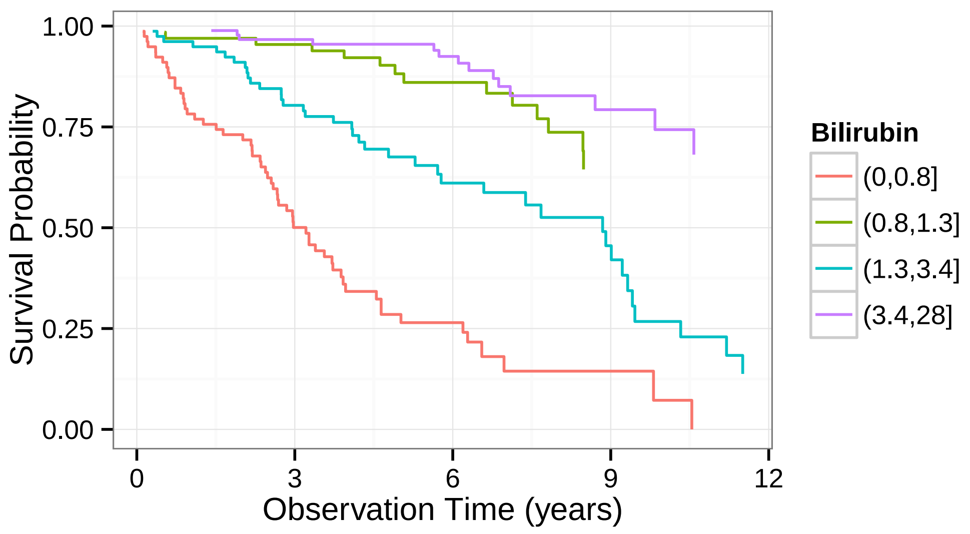

Figure 4: Kaplan–Meier pbc data survival estimates comparing Bilirubin measures. Groups defined in Fleming and Harrington (1991).

3. Random Survival Forest

A Random Forest (Breiman 2001) is grown by *bagging* (Breiman 1996a) a collection of *classification and regression trees* (CART) (Breiman, Friedman, Olshen, and Stone 1984). The method uses a set of B *bootstrap* (Efron and Tibshirani 1994) samples, growing an independent tree model on each sub-sample of the population. Each tree is grown by recursively partitioning the population based on optimization of a *split rule* over the p-dimensional covariate space. At each split, a subset of $m \le p$ candidate variables are tested for the split rule optimization, dividing each node into two daughter nodes. Each daughter node is then split again until the process reaches the *stopping criteria* of either *node purity* or *node member size*, which defines the set of *terminal (unsplit) nodes* for the tree. In regression trees, node impurity is measured by mean squared error, whereas in classification problems, the Gini index is used (Friedman 2000) .

Random Forests sort each training set observation into one unique terminal node per tree. Tree estimates for each observation are constructed at each terminal node, among the terminal node members. The Random Forest estimate for each observation is then calculated by aggregating, averaging (regression) or votes (classification), the terminal node results across the collection of B trees.

Random Forests for survival (Ishwaran 2007; Ishwaran *et al.* 2008) (RF-S) are an extension of Breiman (2001) Random Forests for right censored time to event data. A forest of survival trees is grown using a log-rank splitting rule to select the optimal candidate variables. Survival estimate for each observation are constructed with a Kaplan–Meier (KM) estimator within each terminal node, at each event time.

Random Forests for survival adaptively discover nonlinear effects and interactions and are fully nonparametric. Averaging over trees, with randomization while growing a tree, enables RF-S to approximate complex survival functions, including non-proportional hazards, while maintaining low prediction error. Ishwaran and Kogalur (2010) showed that RF-S is uniformly consistent and that survival forests have a uniform approximating property in finite-sample settings, a property not possessed by individual survival trees.

The **randomForestSRC** `rfsrc` function call grows the forest, determining the type of forest by the response supplied in the `formula` argument. In the following code block, we grow a random forest for survival, by passing a survival (`Surv`) object to the forest. The forest uses all remaining variables in the `pbc.trial` data set to generate survival estimates.

```
R> rfsrc_pbc <- rfsrc(Surv(years, status) ~ .,
+                     data = pbc.trial,
+                     nsplit = 10,
+                     na.action = "na.impute")
R> rfsrc_pbc

                         Sample size: 312
                    Number of deaths: 125
                   Was data imputed: yes
                     Number of trees: 1000
           Minimum terminal node size: 3
       Average no. of terminal nodes: 59.902
```

```
No. of variables tried at each split: 5
              Total no. of variables: 17
                            Analysis: RSF
                              Family: surv
                      Splitting rule: logrank *random*
       Number of random split points: 10
                          Error rate: 16.01%
```

The `print.rfsrc` function returns information on how the random forest was grown. Here the `family="surv"` forest has `ntree=1000` trees (the default `ntree` argument). We used `nsplit=10` random split points to select random split rule, instead of an optimization on each variable at each split for performance reasons.

3.1. Imputation

The **randomForestSRC** package does missing value imputation internally in the `rfsrc` grow call using a modification of the missing data algorithm of Ishwaran *et al.* (2008).

Prior to splitting each node, missing data for the candidate variables are imputed by randomly drawing values from non-missing values also within the node. The purpose of the imputation step is to observations to be sorted into daughter nodes when the split occurs on a variable with missing data, and this imputed value. The split-statistic is calculated on the non-imputed values only. Following a node split, imputed data are reset to missing and the process is repeated until terminal nodes are reached.

Missing data is then imputed using OOB non-missing terminal node data. For integer valued variables and censoring indicators, imputation uses a maximal class rule, whereas continuous variables and survival time use a mean rule.

The proximity matrix from the randomForest is used to update the imputation of the NAs. For continuous predictors, the imputed value is the weighted average of the non-missing obervations, where the weights are the proximities. For categorical predictors, the imputed value is the category with the largest average proximity. This process is iterated iter times.

3.2. Generalization error

One advantage of Random Forests is a built in generalization error estimate. Each bootstrap sample selects approximately 63.2% of the population on average. The remaining 36.8% of observations, the Out-of-Bag (Breiman 1996b) (OOB) sample, can be used as a hold out test set for each tree. An OOB prediction error estimate can be calculated for each observation by predicting the response over the set of trees which were NOT trained with that particular observation. Out-of-Bag prediction error estimates have been shown to be nearly identical to n–fold cross validation estimates (Hastie, Tibshirani, and Friedman 2009). This feature of Random Forests allows us to obtain both model fit and validation in one pass of the algorithm.

Figure 6 shows the predicted survival from an RF-S model, where censored device prediction is colored in blue, and devices experiencing an event are colored in red.

4. Variable Selection

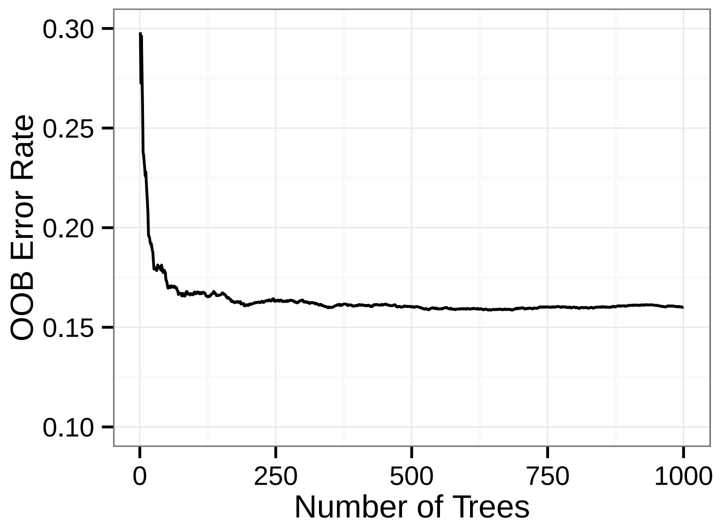

Figure 5: Random forest prediction error estimates as a function of the number of trees in the forest.

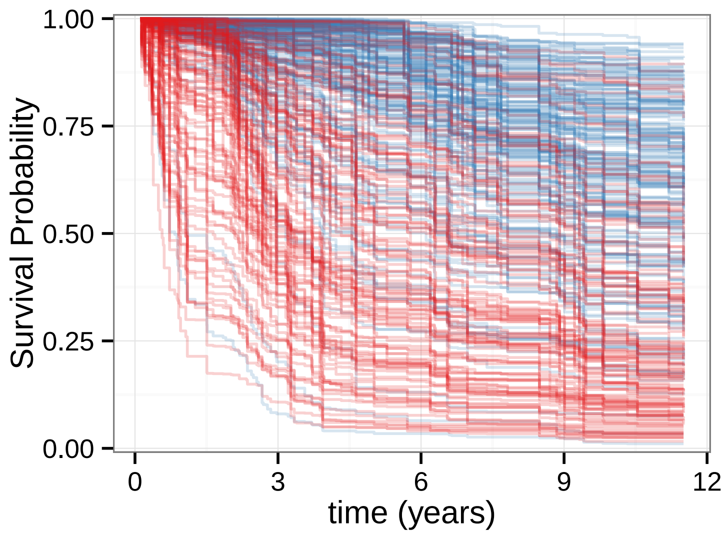

Figure 6: Random forest predicted survival. Blue lines correspond to censored observations, red lines correspond to patients who experienced the event (death).

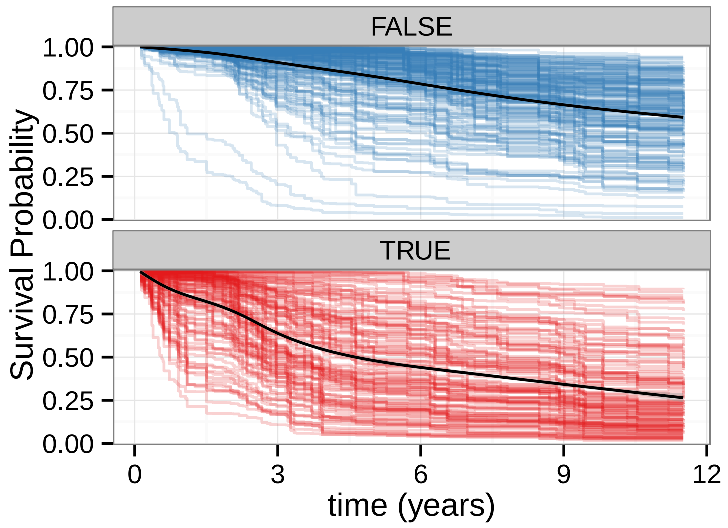

Figure 7: Random forest predicted survival. Split on death event, black loess curve indicates the mean survival estimate within each group.

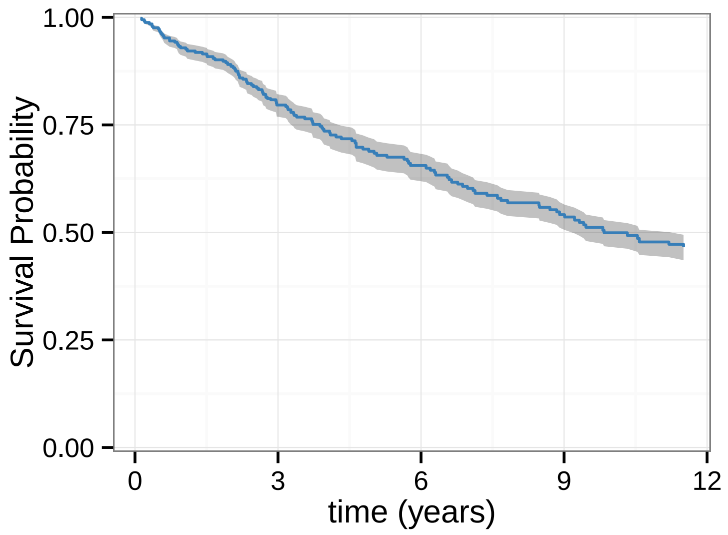

Figure 8: Mean value random forest predicted survival with shaded 95% confidence band.

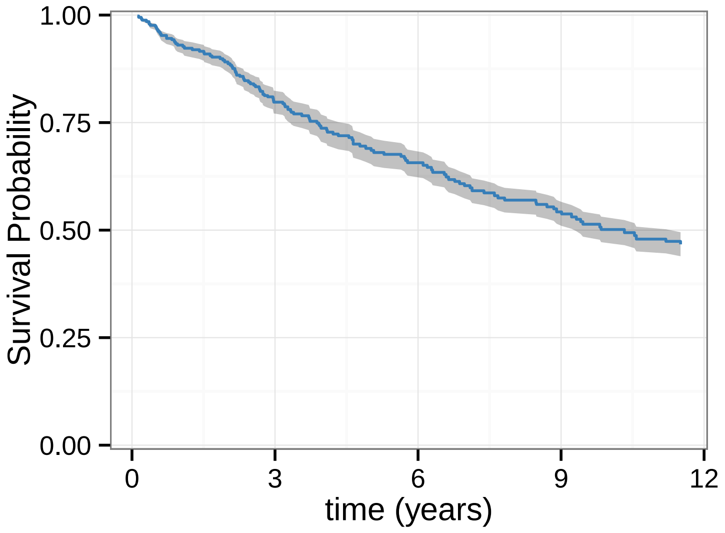

Figure 9: Mean value random forest predicted survival with shaded 95% confidence band.

Unlike in the linear model settings, Random Forests does not require explicitly specify the functional form of the covariates to the response. Instead, we ascertain which variables contribute to the Random Forest estimates by querying the forest for variable usage.

4.1. Variable Importance

Unlike in the linear model settings, Random Forests does not require explicitly specify the functional form of the covariates to the response. Instead, we ascertain which variables contribute to the Random Forest estimates by querying the forest for variable usage.

Variable importance (VIMP) was originally defined in CART using a measure involving surrogate variables (see Chapter 5 of Breiman *et al.* (1984)). The most popular VIMP method to date, adopts a prediction error approach involving "noising-up" a variable. VIMP for a variable x_v is the difference between prediction error when x_v is noised up by permuting its value randomly, compared to prediction error under the original predictor (Breiman 2001; Liaw and Wiener 2002; Ishwaran 2007; Ishwaran *et al.* 2008).

Since VIMP is the absolute difference between prediction errors before and after permutation, a large VIMP value indicates that misspecification of that variable detracts from the predictive accuracy of the forest. VIMP close to zero indicates the variable contributes nothing to predictive accuracy, and negative values indicate the predictive accuracy improves when the variable is mispecified. In the later case, we assume noise is more informative than the variable. As such, we ignore variables with negative and near zero values of VIMP, relying on large positive values to indicate that the predictive power of the forest is dependent on those variables.

In Figure 10, we plot VIMP measures for each of the variables used to grow the forest estimates of Figure 6. Variables are shown in VIMP rank order, largest (op_yr) at the top, to smallest (iv_lospr) at the bottom. In this case, we would focus attention on the top three variables (op_yr (surgical date), ld and devno).

```
R> plot.gg_vimp(rfsrc_pbc, lbls = st.labs) +
+   theme(legend.position = c(.8,.2))+
+   labs(fill="VIMP > 0")+
+   scale_fill_brewer(palette="Set1")
```

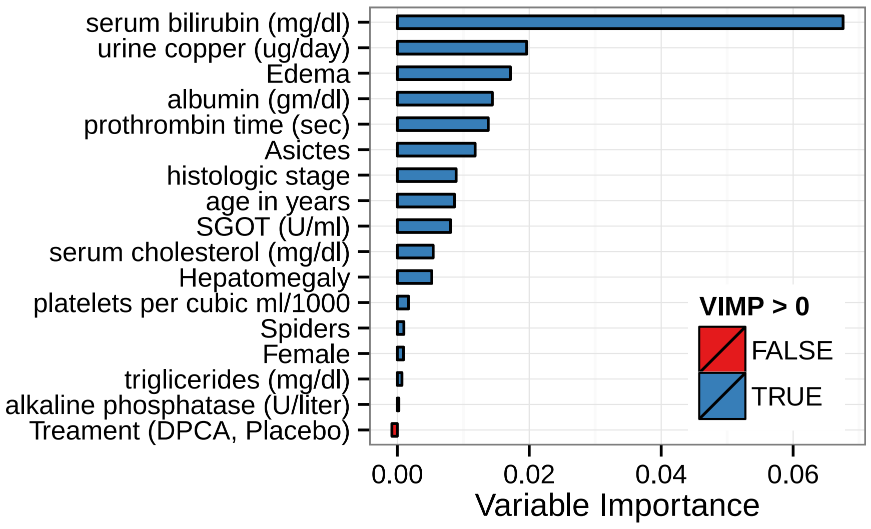

Figure 10: Random forest variable Importance (VIMP). Blue bars indicate important variables (positive VIMP), red indicates noise variables (negative VIMP).

4.2. Minimal Depth

In VIMP, prognostic risk factors are determined by inspection of the forest, ranking the most important variables according to impact on predictive ability of the forest. An alternative method recognizes that most important variables for prediction are those that most frequently split nodes nearest to the trunks of the trees (ie, at the root node) since they partition the largest portions of the population.

Node levels are numbered based on their relative distance to the trunk of the tree (ie. 0, 1, 2). A measure of important risk factors is determined by averaging the depth of first split for each variable over all trees within the forest. Lower values of this measure indicate variables that split larger groups of patients.

The maximal subtree for a variable x is the largest subtree whose root node splits on x. Thus, all parent nodes of x's maximal subtree have nodes that split on variables other than x. The largest maximal subtree possible is the root node. In general, however, there can be more than one maximal subtree for a variable. A maximal subtree may also not exist if there are no splits on the variable. The minimal depth of a maximal subtree (the first order depth) measures predictiveness of a variable x. It equals the shortest distance (the depth) from the root node to the parent node of the maximal subtree (zero is the smallest value possible). The smaller the minimal depth, the more impact x has on prediction. The mean of the minimal depth distribution is used as the threshold value for deciding whether a variable's minimal depth value is small enough for the variable to be classified as strong.

The minimal depth plot of Figure **??** is similar to the VIMP plot in Figure 10, ranking variables from most important at the top (minimal depth measure), to least at the bottom (maximal minimal depth). Since the VIMP and Minimal Depth measures use different criteria, we expect the variable ranking to be slightly different. In this case, minimal depth indicates seven most important variables (op_yr (surgical date), age, ld, ht, wt, iv_lospr (length of stay) and inr). The vertical dashed line indicates the minimal depth threshold where smaller minimal depth values indicate higher importance and larger indicate lower importance.

```
R> varsel_pbc <- var.select(rfsrc_pbc)
R> ggMindepth <- gg_minimal_depth(varsel_pbc, lbls = st.labs)
R> print(ggMindepth)

-----------------------------------------------------------
gg_minimal_depth
model size          : 12
depth threshold     : 5.5811

PE :[1] 16.01
-----------------------------------------------------------

Top variables:
            depth  vimp
bili        1.724 0.068
albumin     2.535 0.014
copper      2.645 0.020
prothrombin 2.841 0.014
chol        3.245 0.005
age         3.466 0.009
edema       3.554 0.017
platelet    3.628 0.002
sgot        3.773 0.008
alk         3.947 0.000
trig        4.233 0.001
stage       4.546 0.009
-----------------------------------------------------------

R> plot(ggMindepth, lbls = st.labs)
```

5. Variable Dependence

Once we have an idea of which variables contribute to the predictive accuracy of the forest, it is useful to get some idea of form of this contribution. We use graphical methods to show the predicted response given dependence on covariates. We can plot the marginal effect of an covariate on the class probability (classification), response (regression), mortality (survival), or the expected years lost (competing risk) for a RF analysis. We plot the ensemble predicted value on the vertical axis and covariates along the horizontal axis.

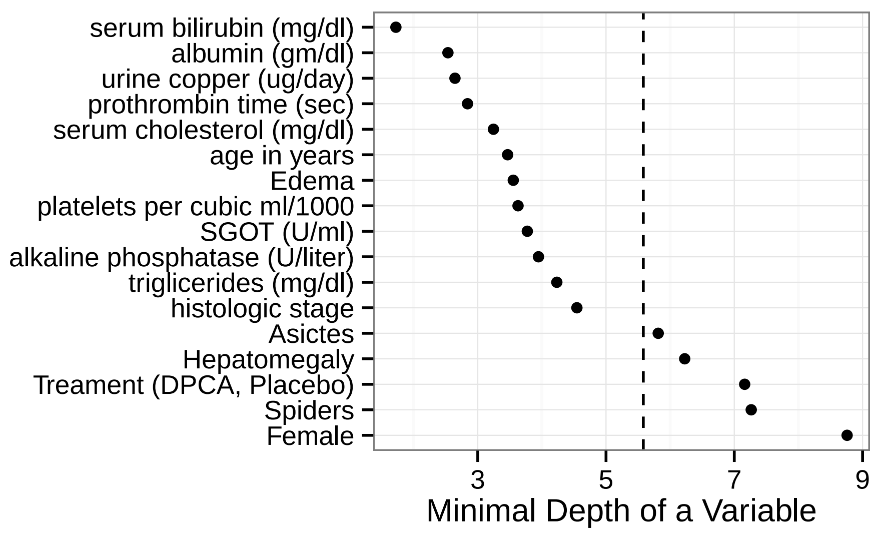

Figure 11: Minimal Depth variable selection. Low minimal depth indicates important variables. The dashed line is the threshold of maximum value for variable selection.

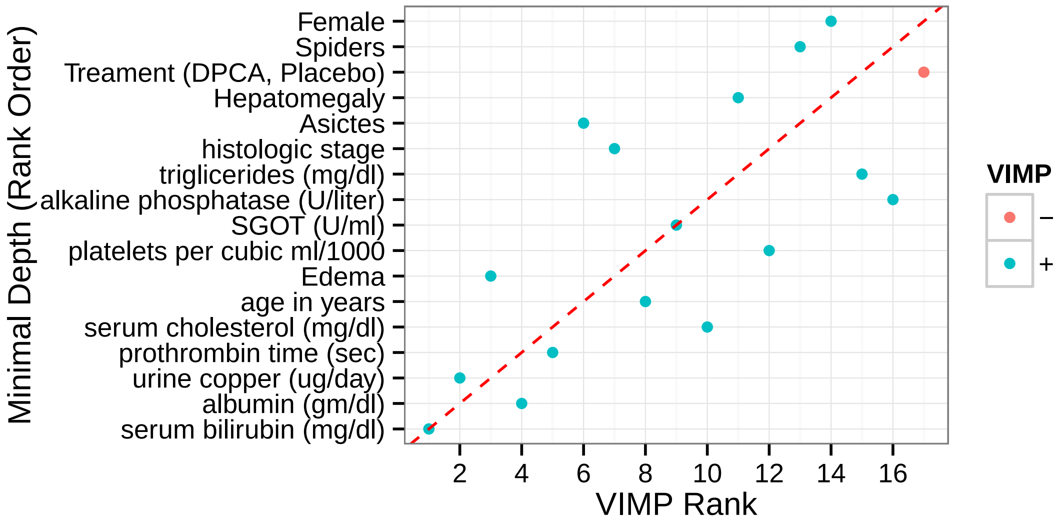

Figure 12: Comparing Minimal Depth and Vimp rankings. Points on the red dashed line are ranked equivalently, points below have higher VIMP, those above have higher minimal depth ranking. Variables are colored by the sign of the VIMP measure.

5.1. Marginal Dependence

Marginal variable dependence plots the predicted response as a function of the covariate, showing each subject as a point on the plot. For classification and regression, this is straight forward predicting the response. In survival settings, we must account for the additional dimension of time. In this case, we plot the response at a specific time point of interest, for example survival at three months shown by the vertical dashed line in Figure 13. We take the predicted value of each curve at that time, and plot that against the covariate value for that observations, shown in Figure **??**. Again censored cases are shown in blue circles, events are indicated by the red ”x” symbols. Each predicted point is dependent on the full combination of all other covariates, not only on the covariate displayed in the dependence plot, so interpretation of these variable dependence plots can only be in general terms. The smooth loess line (Cleveland 1981; Cleveland and Devlin 1988) indicates the trend of the prediction over surgical date progression.

```
R> ggRFsrc +
+    geom_vline(aes(xintercept = c(1, 3)), linetype = "dashed") +
+    coord_cartesian(x = c(0, 4))
```

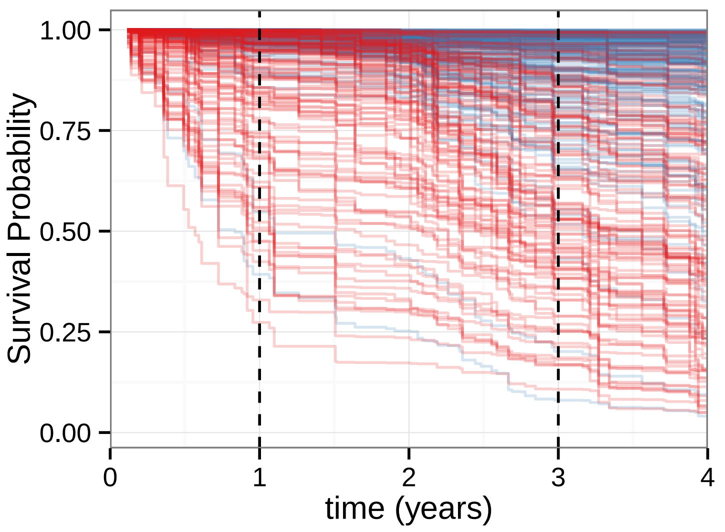

Figure 13: Random forest OOB predicted patient survival. Red curves correspond to patients which have died, blue corresponds to alive (or censored) cases. Vertical dashed lines indicate the 1 and 3 year survival estimates.

```
R> xvar <- varsel_pbc$topvars[1:7]
R>
R> ggrf <- gg_variable(rfsrc_pbc, time = c(1, 3),
+                          time.labels = c("1 Year", "3 Years"))
R>
R> plot(ggrf, xvar = xvar[1], se=.95, alpha=.3) +
+   labs(y = "Survival") +
+   theme(legend.position = "none") +
+   scale_color_manual(values = strCol, labels = event.labels) +
+   scale_shape_manual(values = event.marks, labels = event.labels)
```

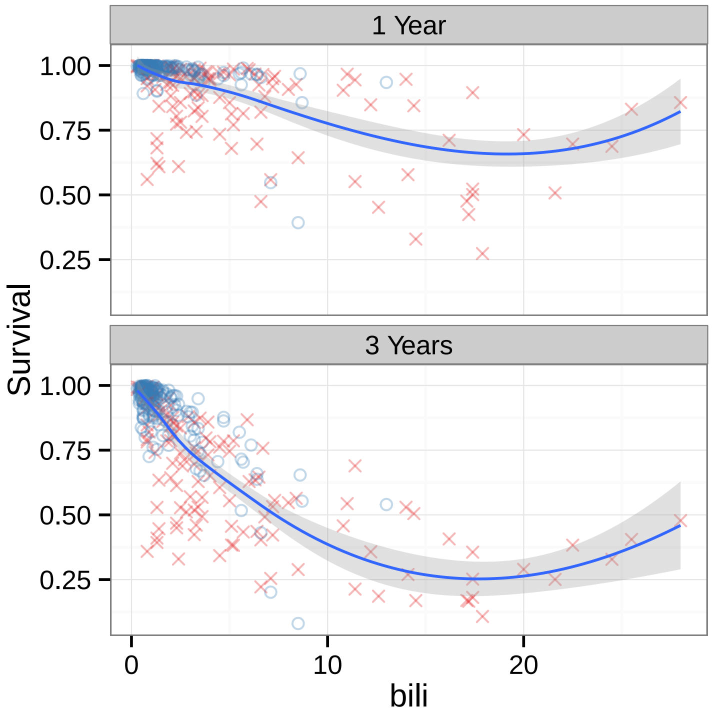

Figure 14: Bilirubin variable dependence at 1 and 3 years. Individual cases are marked with blue circles (alive or censored) and red xs (dead). Loess smooth curve with shaded 95% confidence band indicates the survival trend with increasing bilirubin.

```
R> plot(ggrf, xvar = xvar[-c(1,7)], panel = TRUE,
+       se=FALSE, alpha=.3,
+       method="glm", formula=y~poly(x,2)) +
+   labs(y = "Survival") +
+   theme(legend.position = "none") +
+   scale_color_manual(values = strCol, labels = event.labels) +
+   scale_shape_manual(values = event.marks, labels = event.labels)+
+   coord_cartesian(y=c(-.1,1.02))

R> plot(ggrf, xvar = "edema", notch=TRUE, alpha=.5) +
+   labs(y = "Survival") +
+   theme(legend.position = "none") +
+   scale_color_manual(values = strCol, labels = event.labels) +
+   scale_shape_manual(values = event.marks, labels = event.labels)+
+   coord_cartesian(y=c(-.1,1.02))
```

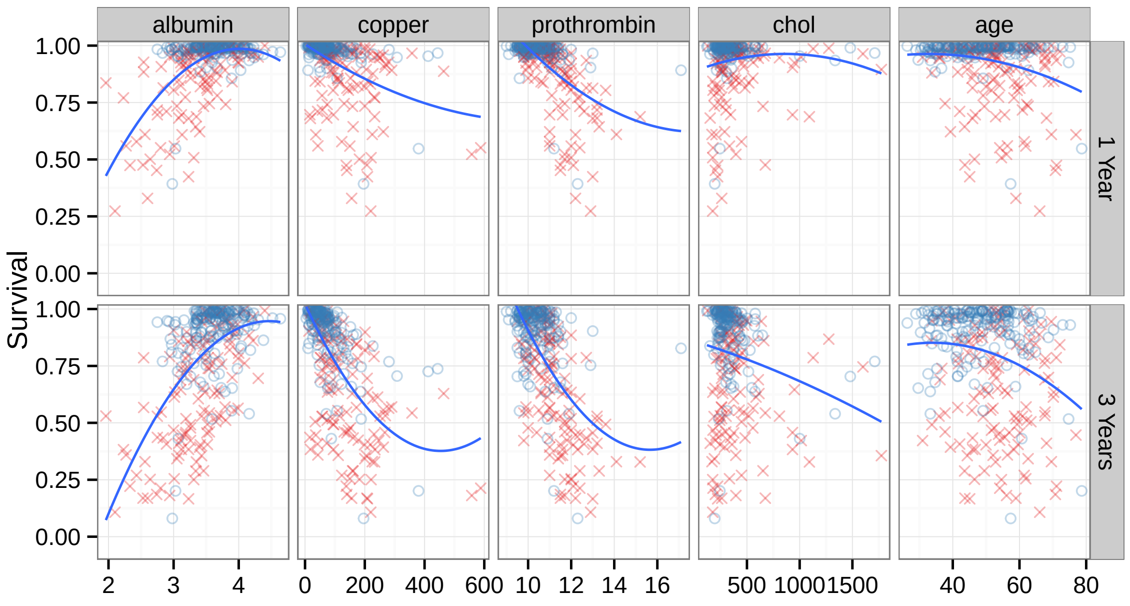

Figure 15: Variable dependence plots at 1 and 3 years for continuous variables age, albumin, copper and prothrombin. Individual cases are marked with blue circles (alive or censored) and red xs (dead). Loess smooth curve indicates the survival trend with increasing variable value.

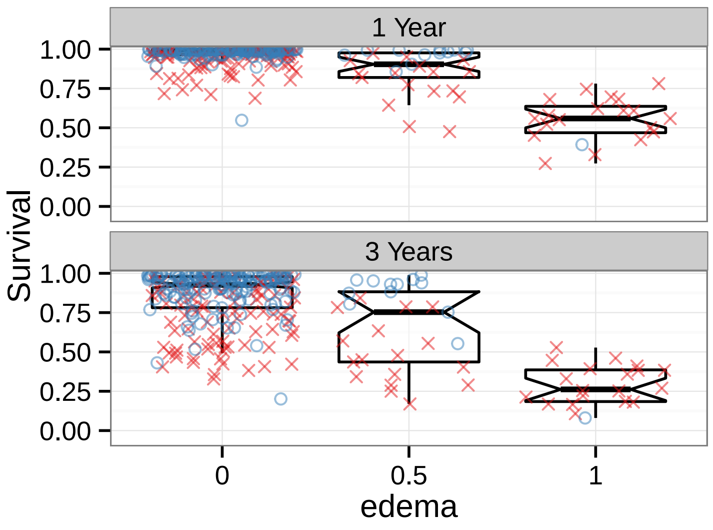

Figure 16: Variable dependence plots at 1 and 3 years for categorical edema variable. Individual cases are marked with blue circles (alive or censored) and red xs (dead). Boxes indicate distributional properties of observations in each group.

5.2. Partial Dependence

Partial dependence plots are a risk adjusted alternative to marginal variable dependence. Partial plots are generated by integrating out the effects of variables beside the covariate of interest. The figures are constructed by selecting points evenly spaced along the distribution of the X variable. For each of these values (X = x), we calculate the average Random Forest prediction over all other covariates in X by (1).

$$\tilde{f}(x) = \frac{1}{n}\sum_{i=1}^{n} \hat{f}(x, x_{i,o}), \tag{1}$$

where $\hat{f}$ is the predicted response from the random forest and $x_{i,o}$ is the value for all other covariates other than $X = x$ for the observation i (Friedman 2000). Partial dependence plots in time to event settings are shown at specific time points, similar to variable dependence.

Figure 17 shows the partial dependence of three month survival on bilirubin.

```
R> # Calculate the 1, 3 and 5 year partial dependence
R> partial_pbc <- lapply(c(1,3,5), function(tm){
+   plot.variable(rfsrc_pbc, surv.type = "surv",
+                 time = tm,
+                 xvar.names = xvar, partial = TRUE,
+                 show.plots = FALSE)
+   })

R> # Convert all partial plots to gg_partial objects
R> gg_dta <- lapply(partial_pbc, gg_partial)
R>
R> # Combine the objects to get multiple time curves
R> # along variables on a single figure.
R> pbc_ggpart <- combine.gg_partial(gg_dta[[1]],gg_dta[[2]],
+                                  lbls = c("1 Year", "3 Years"))
R>
R> plot(pbc_ggpart[["bili"]], se = FALSE) +
+   theme(legend.position = c(.8, .5)) +
+   labs(y = "Survival",
+        x = st.labs["bili"],
+        color="Time", shape="Time")+
+   scale_color_brewer(palette="Set1")
```

Non-proportional hazards are evident in Figure 17.

```
R> # Create a temporary holder and remove the bilirubin and edema data
R> ggpart <- pbc_ggpart
R> ggpart$edema <- NULL
R>
R> # Panel plot the remainder.
R> plot(ggpart, se = FALSE, panel = TRUE) +
```

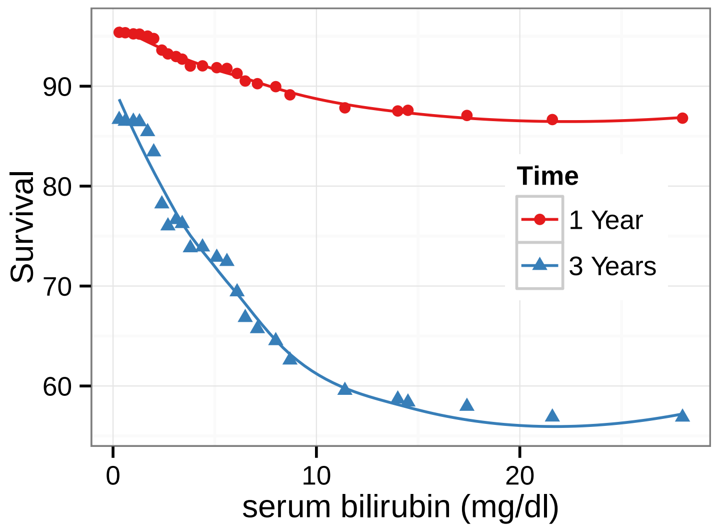

Figure 17: Partial dependence plot of (risk adjusted) predicted survival probability as a function of serum bilirubin at 1 year (red circle) and 3 years (blue triangle). Loess smooth curves indicates the trend.

```
+    labs(x = "", y = "Survival",color="Time", shape="Time") +
+    scale_color_brewer(palette="Set1")
```

```
R> plot(pbc_ggpart$edema, notch=TRUE, alpha=.3, outlier.shape = NA) +
+    labs(x = "Edema", y = "Survival (%)")+
+    scale_color_brewer(palette="Set1")+
+    facet_grid(~group)+
+    theme(legend.position="none")
```

6. Variable Interactions

Using the different variable dependence measures, we can calculate pairwise interactions for any pair of variables. Minimal depth is calculated as the maximal subtree using the normalized minimal depth of variable i relative to the root node (normalized with respect to the size of the tree). For interactions, we calculate the maximal subtree interaction measure as the normalized minimal depth of a variable j with respect to the maximal subtree for variable i (normalized with respect to the size of i's maximal subtree) (Ishwaran *et al.* 2010; Ishwaran, Kogalur, Chen, and Minn 2011).

```
R> interaction_pbc <- find.interaction(rfsrc_pbc)
R>
R> ggint <- gg_interaction(interaction_pbc)
R> plot(ggint, xvar = "bili") +
+    labs(y = "Interactive Minimal Depth")
```

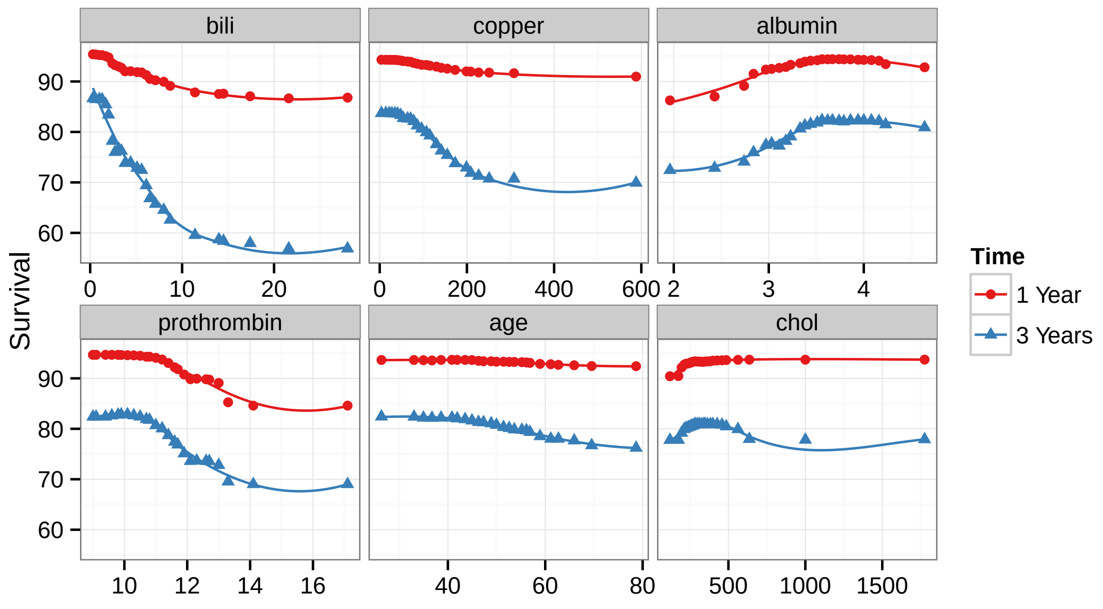

Figure 18: Partial dependence plot of (risk adjusted) predicted survival probability as a function continuous variables prothrombin time, albumin, age and urin copper at 1 year (red circle) and 3 years (blue triangle).

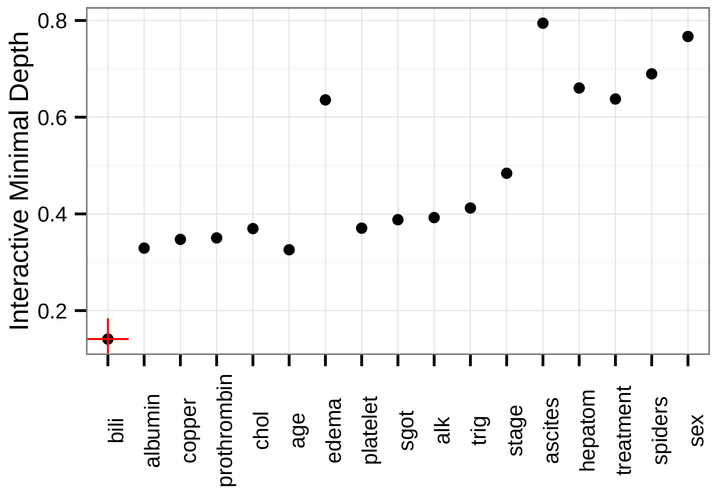

Figure 19: Minimal depth variable interaction with bilirubin (marked with red cross). Higher values indicate lower interactivity with target variable.

Measuring interactions with minimal depth results a $p \times p$ matrix of interaction measures, with smaller diagonal measures relative to the root node, and off diagonal measures of pair-wise interaction. We expect the covariate with smallest minimal depth to have the highest interactive depth measures, so viewed alone may not be as informative as looking at other interactive depth plots. Figure 20 combines the remaining top ranked minimal depth measures for comparison.

```
R> plot(gg_interaction(interaction_pbc), xvar = xvar[-1]) +
+    labs(y = "Interactive Minimal Depth") +
+    theme(legend.position = "none")
```

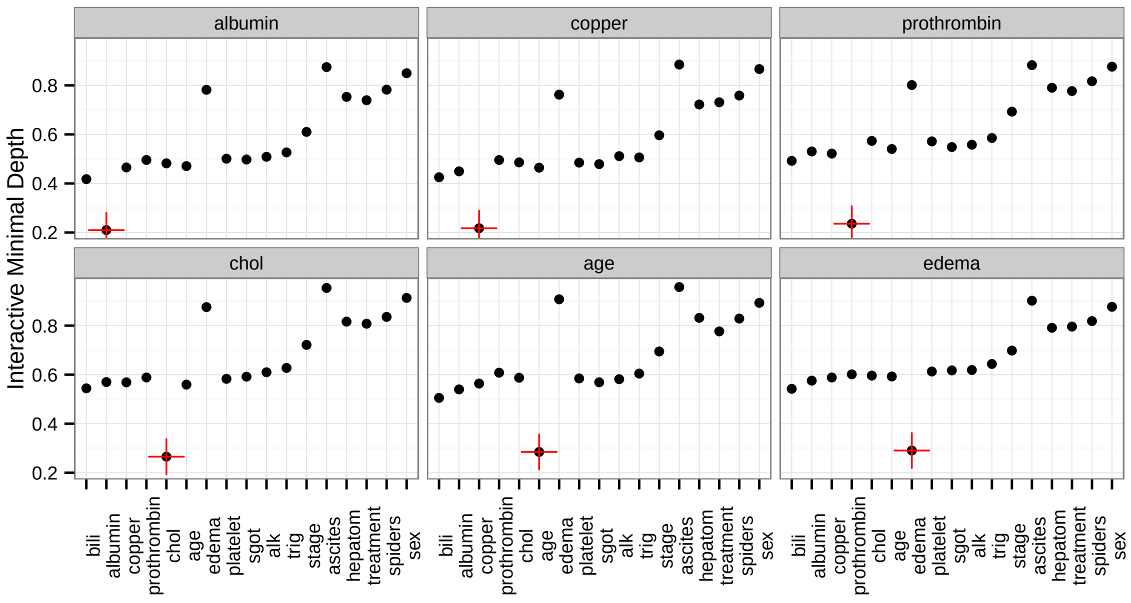

Figure 20: Minimal depth variable interaction panel with prothrombin time, albumin, urine copper and edema. Higher values indicate lower interactivity with target variable.

7. Conditional Dependence Plots

By plotting the resulting interaction measures for each variable (Figure 19), we can detect the "most interactive" pairs, and develop conditional plots Chambers (1992); Cleveland (1993). These plots are similar to stratified results, arranged in a set of panels by the interactive variable of interest.

Interactions with categorical data are more straight forward, and can be generated directly from variable dependence plots. Recall the 1 year variable dependence for Billirubin, shown in Figure 21.

```
R> ggvar <- gg_variable(rfsrc_pbc, time = 1)
R> ggvar$stage <- paste("stage=", ggvar$stage, sep="")
```

```
R>
R> var_dep <- plot(ggvar, xvar = "bili", smooth = TRUE,
+                  method = "loess", span=1.5,alpha = .5, se = FALSE) +
+   labs(y = "Survival",
+        x = st.labs["bili"]) +
+   theme(legend.position = "none") +
+   scale_color_manual(values = strCol, labels = event.labels) +
+   scale_shape_manual(values = event.marks, labels = event.labels)
R>
R> show(var_dep)
```

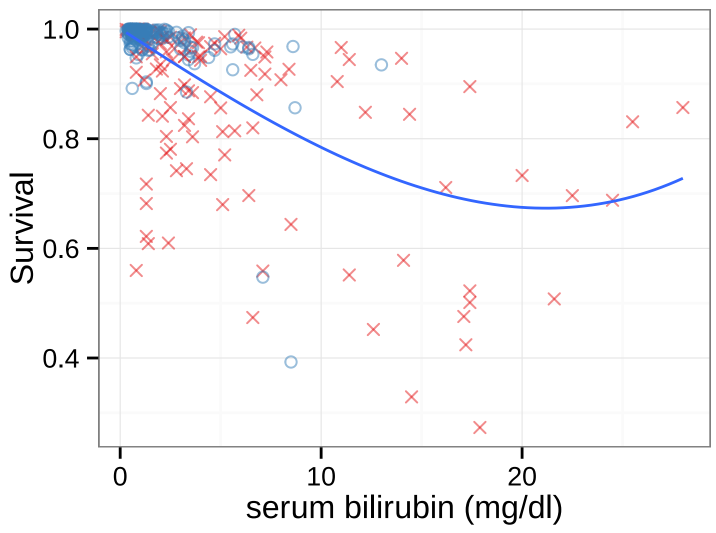

Figure 21: Variable dependence plot. Survival at 1 year against bilirubin. Individual cases are marked with blue circles (alive or censored) and red x (dead). Loess smooth curve indicates the trend as bilirubin increases.

We can view the conditional dependence of survival against bilirubin, versus other categorical covariates, say treatment (binary) and stage (categorical), by adding a facet argument.

```
R> var_dep +
+   facet_grid(treatment~stage)
```

Interactions with continuous variables requires stratification at some level.

7.1. Albumin

```
R> # albumin_grp <- cut(rfsrc_pbc$xvar$albumin, breaks=c(0,seq(3,3.5,.5),5))
R> albumin_grp <- cut(rfsrc_pbc$xvar$albumin, breaks=6)
R> ggvar$albumin_grp <- paste("albumin=",albumin_grp, sep="")
R>
R> var_dep <- plot(ggvar, xvar = "bili", smooth = TRUE,
```

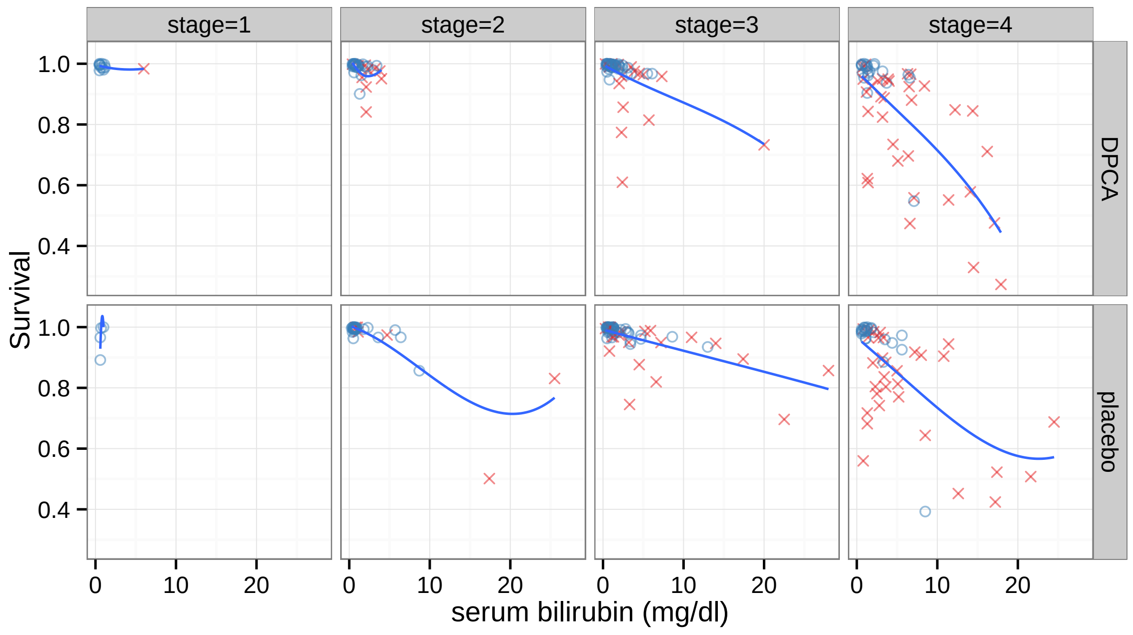

Figure 22: Variable dependence coplot. Survival at 1 year against bilirubin, stratified by treatment and histological stage.

```
+                  method = "loess", span=1.5,alpha = .5, se = FALSE) +
+   labs(y = "Survival", x = st.labs["bili"]) +
+   theme(legend.position = "none") +
+   scale_color_manual(values = strCol, labels = event.labels) +
+   scale_shape_manual(values = event.marks, labels = event.labels)+
+   facet_wrap(~albumin_grp)
R>
R> var_dep
```

```
R> data(partial_coplot_pbc, package="ggRandomForests")
R> ggpl <- ggplot(partial_coplot_pbc, aes(x=bili, y=yhat,
+                                           shape=group,
+                                           color=group))+
+   geom_point()+geom_smooth(se=FALSE)+
+   labs(x=st.labs["bili"], y="Survival 1 year",
+        color="Albumin", shape="Albumin")+
+   scale_color_brewer(palette="Set1")
R> ggpl
```

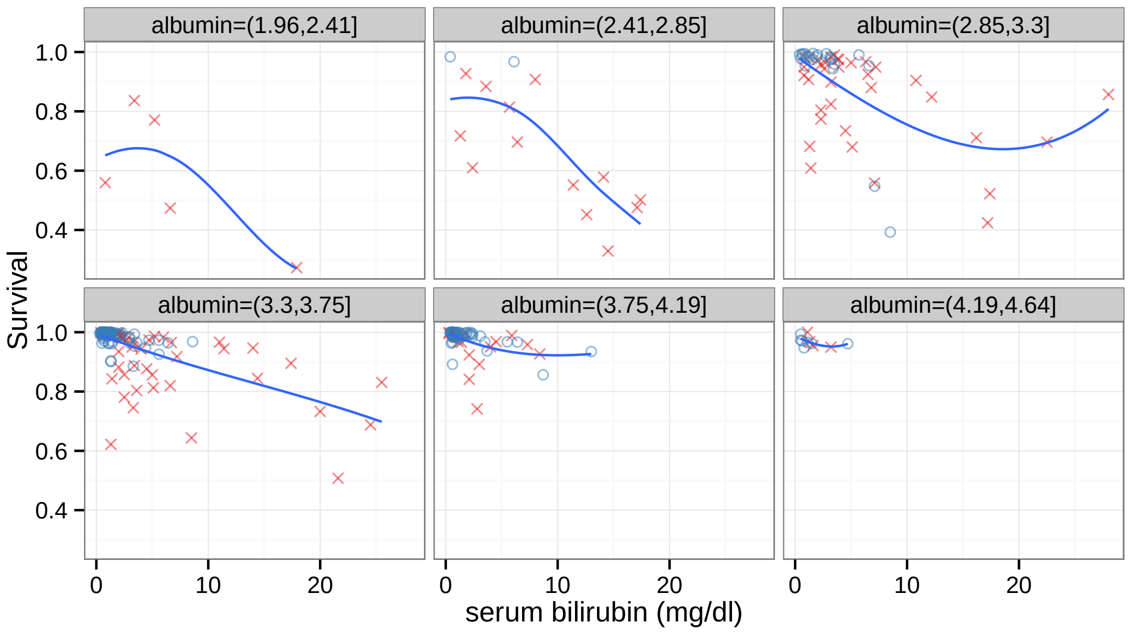

Figure 23: Variable dependence coplot. Survival at 1 year against bilirubin, stratified by continous variable albumin.

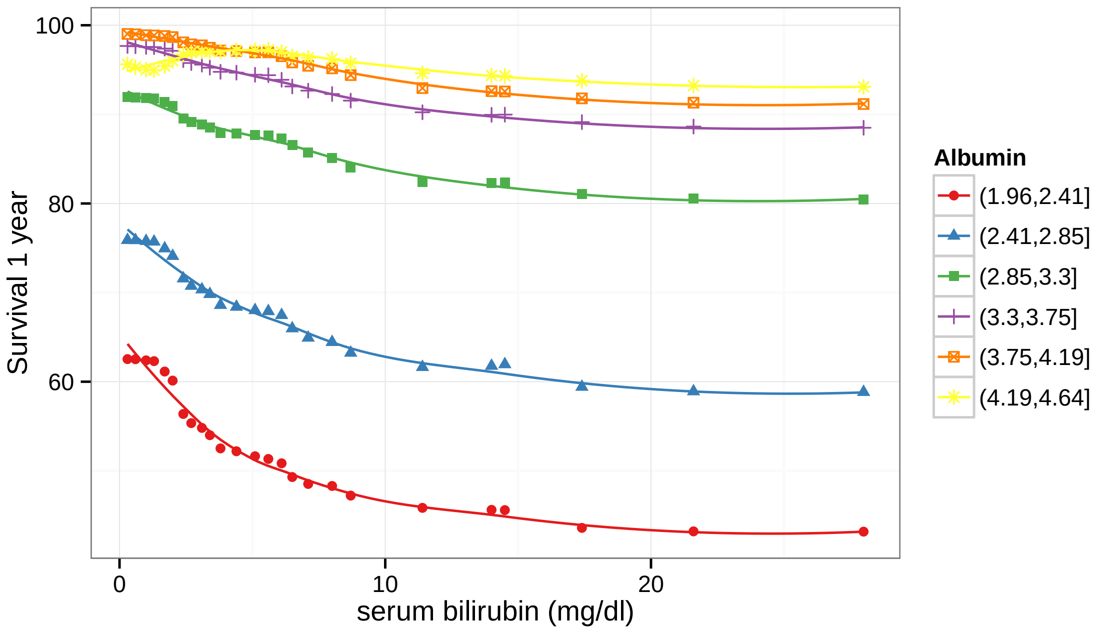

Figure 24: Partial (risk adjusted) variable dependence coplot. Survival at 1 year against bilirubin, stratified by albumin groups. Points mark risk adjusted estimates, loess smooth indicates predicted trend within each age group as a function of bilirubin.

8. Conclusion

References

Breiman L (1996a). "Bagging predictors." *Machine Learning*, **26**, 123–140.

Breiman L (1996b). "Out–Of–Bag Estimation." *Technical report*, Statistics Department, University of California,Berkeley, CA. 94708. URL `ftp://ftp.stat.berkeley.edu/pub/users/breiman/OOBestimation.ps.Z`.

Breiman L (2001). "Random Forests." *Machine Learning*, **45**(1), 5–32.

Breiman L, Friedman JH, Olshen R, Stone C (1984). *Classification and Regression Trees.* Wadsworth and Brooks, Monterey, CA.

Chambers JM (1992). *Statistical Models in S.* Wadsworth & Brooks/Cole.

Cleveland WS (1981). "LOWESS: A program for smoothing scatterplots by robust locally weighted regression." *The American Statistician*, **35**(1), 54.

Cleveland WS (1993). *Visualizing Data.* Summit Press.

Cleveland WS, Devlin SJ (1988). "Locally-Weighted Regression: An Approach to Regression Analysis by Local Fitting." *Journal of the American Statistical Association*, **83**(403), 596–610.

Efron B, Tibshirani R (1994). *An Introduction to the Bootstrap.* Chapman & Hall/CRC. ISBN 0412042312.

Fleming TR, Harrington DP (1991). *Counting processes and survival analysis.* Wiley, New York.

Friedman JH (2000). "Greedy Function Approximation: A Gradient Boosting Machine." *Annals of Statistics*, **29**, 1189–1232.

Hastie T, Tibshirani R, Friedman JH (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2 edition. Springer. ISBN 978-0-387-84857-0.

Ishwaran H (2007). "Variable importance in binary regression trees and forests." *Electronic Journal of Statistics*, **1**, 519–537.

Ishwaran H, Kogalur UB (2007). "Random survival forests for R." *R News*, **7**, 25–31.

Ishwaran H, Kogalur UB (2010). "Consistency of random survival forests." *Statistics and Probability Letters*, **80**, 1056–1064.

Ishwaran H, Kogalur UB (2014). "Random Forests for Survival, Regression and Classification (RF-SRC), R package version 1.5.5."

Ishwaran H, Kogalur UB, Blackstone EH, Lauer MS (2008). "Random survival forests." *The Annals of Applied Statistics*, **2**(3), 841–860.

Ishwaran H, Kogalur UB, Chen X, Minn AJ (2011). "Random survival forests for high-dimensional data." *Statist. Anal. Data Mining*, **4**, 115–132.

Ishwaran H, Kogalur UB, Gorodeski EZ, Minn AJ, Lauer MS (2010). "High-dimensional variable selection for survival data." *J. Amer. Statist. Assoc.*, **105**, 205–217.

Liaw A, Wiener M (2002). "Classification and Regression by randomForest." *R News*, **2**(3), 18–22.

Tukey JW (1977). *Exploratory Data Analysis*. Pearson.

Wickham H (2009). *ggplot2: elegant graphics for data analysis*. Springer New York. ISBN 978-0-387-98140-6.

Affiliation:

John Ehrlinger
Quantitative Health Sciences
Lerner Research Institute
Cleveland Clinic
9500 Euclid Ave
Cleveland, Ohio 44195
E-mail: john.ehrlinger@gmail.com
URL: http://www.lerner.ccf.org/qhs/people/ehrlinj/