Accuracy and Precision of Time-to-Event Models with Flexible Dimensionality

Brendan Cook, Vikenty Mikheev, Bibekananda Mishra, Negar Orangi-Fard, Jake Roberts

Mentor: Jonathan Hill

July 2019

1 Executive Summary

Using the the ITM 21st longevity data set, our objective was to design a 5 question questionnaire that would accurately predict the life outcomes of individuals who would take the questionnaire. Such a questionnaire would allow for a company such as ITM 21 to accurately price the life insurance policies of their clientele. According to the existing literature in survival analysis, the idea is to fit each individual with a particular survival function that gives the probability such an individual will survive until a given time t (here, t is the time since beginning of observation; not age of the individual). A few difficulties arise with this sort of analysis.

The main problem is to determine what information is most influential to affecting the survival function of an individual. In this data set, there are 125 different covariates (predictors) associated to each person in the study. In order to design a questionnaire that can most accurately determine an individuals probability of surviving, these most influential predictors must be determined somehow.

Another problem is the simple fact that most of the the people in this study (about 3/4) are still alive. This distinction makes using techniques in standard regression impractical since most of the people in the data have no conclusive time of event (in this case, the event being that they die). This extra dimension of data demands for a more sophisticated approach when designing a model for this data set.

Over the passed week, our group took two different approaches to solving this problem. The first approach was using variable selection, and the second approach was to use branching methods. The variable selection method is equivalent to asking everyone the same questions that identify those variables that we had deemed as most influential. In many ways, due to the lack of flexibility of this model, this approach was used as more of a bench mark to compare with than as a final result of our findings.

The other approach we explored is the approach of using branching methods to design a questionnaire. Unlike variable selection, branching methods design a questionnaire in which the questions that are provided depend on the answers given to previous questions. This extra dimension of flexibility makes this approach more attractive than variable selection, but because of the sophistication of this approach, implementation of these branching methods posed a major challenge for our group.

Ultimately, the the more sophisticated approach coming from branching methods worked significantly better than the standard approach of variable selection. Performance of these models were measured using the standard C-index obtained from a holdout set of 15,000 individuals.

In conclusion, by using a branching method approach, a 5 question questionnair can be designed in order to accurately asses an individuals probability of survival. This assesment is signifigantly better than using the standard same 5 questions for everyone that comes from a variable selection approach. We have not explored questionnairs involving more questions or to what effect more questions would have on improving the C-index of the determined model. The benefit of these questionnairs allows for insurance companies such as ITM 21 to accurately assess their clientele in a cost effective manner and provide an accurate prediction for their probability of survival.

2 Introduction

In our project, we have been given the task of designing a 5 question questionnaire under which we could most accurately predict the probability an individual will survive. We have been provided with the ITM 21st longevity dataset to design such a questionnaire. In this dataset, there are 48,074 lives with their mortality outcomes. Since many (in fact most) in this data set are still alive, it is not sensible to use standard methods of regression analysis in determining such a questionnaire. For each person in this data set, 125 predictors are recorded including exact age, gender, smoker, and BMI. Our objective is to identify the most crucial of this information using the questionnaire under which we can then accurately asses and individuals probability of surviving over some duration

3 Preliminaries

For a given individual in our data set, we would like to be able to provide a survival function $S(t)$ which is defined to be the probability that that individual will survive until time t after beginning observation. More precisely,

$$S(t) = P(T > t)$$

Where T is the random variable associated with this persons life expectancy. In terms of integrals,

$$S(t) = \int_t^\infty f(u)\ du$$

where $f(u)$ is the pdf associated to the random variable T. The pdf $f(u)$ is related to the hazards function $\lambda(t)$ defined by

$$\lambda(t) = \frac{f(t)}{S(t)}.$$

We attempt to model the hazard function $\lambda(t)$ using the Cox proportional hazards (CPH) model defined by

$$h(t \mid X) = h_0(t)e^{\beta^T X} \approx \lambda(t).$$

Here, $h(t \mid X)$ is a model for the hazard function $\lambda(t)$ associated to some particular person, and X is the vector of all 125 features of this person. Most features, such as smoking, have a binary value. Other features, such as exact age however, are continuous quantities. The vector β corresponds to optimal values obtained using a Bayesian approach to analyzing the data in which a defined likelihood function is maximized. The function $h_0(t)$ is the baseline function determined by the Kaplan-Meier estimator of the survival function, and the coefficient $e^{\beta^T X}$ is the Cox hazard proportion for that individual.

The approach of variable selection attempts to solve our problem by selecting the most influential features in the vector X. This approach with results are explained in the next section.

4 Variable selection approach

One well-known method to select variables for a model is backward stepwise selection by p-values of the coefficients. We run one down to 7 predictors: 6013, 8020, 38106, 38107, 35102, ExactAge, CertGender_M. The C-Index on the test data was 0.72976.

The observation of this selection process showed that half of the selected variables had extremely small p-values (less than 10^{-5}). That pushed us to the idea that maybe other method of selection must be chosen for these predictors.

Previously, ITM 21st have already used selection by absolute value of coefficients (taking away 20% smallest predictors on each round of selection). The achieved C-index was 0.72053. We decided to combine these two methods: On each round of selection take away one predictor with highest p-value if it is greater than 10^{-3}, but if the largest p-value smaller than 10^{-3} we select out the coefficient with smallest absolute value. This approach is fair only if all coefficient have the same range of absolute values. That is why normalization of BMI and ExactAge was necessary, which was easily done by

$$\text{NormalizedBMI}_i = \frac{\text{BMI}_i - \text{mean(BMI)}}{\max_i |\text{BMI}_i - \text{mean(BMI)}|}$$

and

$$\text{NormalizedExactAge}_i = \frac{\text{ExactAge}_i - \text{mean(ExactAge)}}{\max_i |\text{ExactAge}_i - \text{mean(ExactAge)}|},$$

where $i = \overline{1, \text{len}(data)}$.

We run this hybrid selection down 31 predictors and added interactions of these 31 with ExactAge. Then we run it again down to 6 predictors: 38105, 3005, 38107, 39115, NormalizedExactAge, AgeIntSmoker, AgeInt39115. Then we added CertGender_M back since this predictor as well as ExactAge will be provided in anyway. The C-Index on test data was 0.71008.

The observation that C-Index on the model with hybrid selection is worse than one on simple p-value was surprising. Is this a nature of data or theoretical issue with absolute value selection or unfortunate property of C-Index? We don't know yet. This place requires more study.

5 Branching Approaches

5.1 Decision Tree approach

A decision tree is a decision support tool that uses a tree like graph or model of decisions and their possible consequences including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

Desion Tree approach comes under the strategy of "branching model approach". In this approach, we first built a *master model* through Cox Proportional Hazard model involving all the variables. This model has unsurprisingly high C-index(C-index= 0.75270). Then, we sort the whole data set according to the 'mortality risk factor' associated with each individual. Then, we made 10 bins where each bin corresponds to tenth-quantile of the sorted data. We record the associated bin number in a separate column to the original data matrix. After removing the columns of 'death' and 'duration', the plan was to fit the whole data to a classification tree model where the tree would provide the bin number associated to each person. Then the decision tree model would be pruned to depth 5 to build a 5-question questionnaire. Then, the master model would be used to corresponding to the predicted bins of the final output to come up with correct distributions for each person. We made progress toward implementing the decision tree, but we could not test the final model. The libraries we used in R are "survival" for CPH model, "Hmisc" for computing the C-index, and "tree" and "ISLR" for building the tree.

5.2 Survival Tree approach

A survival tree is a variant of decision tree suitable for survival analysis that uses a log-rank hypothesis test for node splitting and gives a survival function as output. Our approach was to train a survival tree using the "rpart" and "survival" libraries in R, and then prune the tree down to the correct depth for a 5-question survey. Since we assumed to know the age and gender of any subject filling out the questionnaire, these were not counted toward the 5 questions. This consideration led to increased difficulty in pruning, since pruning the tree to a depth of 5 can result in a questionnaire with less than 5 questions, due to age and gender splits. We handled this issue by examining the node numbering system used by the rpart library, and writing a function to automatically flag those nodes that must be pruned. After pruning our survival tree model to the correct 5-question depth, we then used split complexity pruning to remove additional nodes if necessary to prevent overfitting. We also made some progress toward selecting the optimal cost parameters based on cross-validation, but this was not fully completed.

6 Results and Conclusion

Model	C-index
Baseline Model (age/gender only)	0.70839
5-variable baseline model	0.72053
Survival Forest	0.81303
CPH 20%round abs-val selection	0.72053
CPH Stepwise p-val selection	0.72976
CPH Stepwise hypbrid selection	0.71008
Survival Tree	0.73584
Master model for decision tree	0.75270

As we observe in the table above, our Survival tree and decision tree branching models achieve better predictive performance as compared to both the baseline model and the 5-question variable selection approaches. Given more time and proper tuning of hyperparameters, we believe the performance of these models could be improved even further.

With three observed experiments of CPH with backward selection (20% of abs-value in each round, p-value stepwise, hybrid stepwise), we may conclude that for this problem using C-index as criteria of quality of the model the selection by p-values gives better results than involving absolute value selection.

7 Appendix

7.1 Decision Tree approach code

```
```{r}
library("survival")
library("ISLR")
library("tree")
library("Hmisc")
library("dplyr")
```

```{r}
df <- ITM2019BootcampDataCleaned
names(df)[2]<-"time"
names(df)[3]<-"status"
```
```{r}
df1 = sort(sample(nrow(df), nrow(df)*0.7))
train_df <- df[df1,]
```
```

```
test_df <- df[-df1,]
```
```{r}
coxmodel <- coxph(formula=Surv(time, status) ~ .-X1-YearsFromUW, data = train_df)
```
```{r}
pretrain<-predict(coxmodel, train_df, type="risk")
```
```{r}
pretest<-predict(coxmodel, test_df, type="risk")
test_df$pretest <- pretest
```
```{r}
testFit<-Surv(time=test_df$time,event=test_df$status)
rcorr.cens(1-test_df$pretest, S=testFit)
```
```{r}
holoutset<-ITM2019BootcampDataHoldoutNoOutcomes
holoutset$BMI[is.na(holoutset$BMI)]<- 27.19
```
```{r}
spre<-sort(pre, decreasing = FALSE)
```
```{r}
q<-quantile(spre,seq(0,1,0.1))
```
```{r}
train_df<-mutate("prediction"=pre,train_df)
```
```{r}
computeBin <- function(score) {
  for (i in 2:11) {
    if (score <= q[i]) {
      return(i-1)
    }
  }
}
```
```{r}
bin_no <-c()
for (i in 1:nrow(train_df)){
  bin_no<-append(bin_no,computeBin(train_df$prediction[i]))
}
```
```{r}
```

```
train_df<-mutate("bin"=bin_no,train_df)
```
```
```{r}
# We did not manage to fit the tree
train_df$bin <- as.factor(train_df$bin)
tree.fit <- tree(formula = bin ~.-X1-YearsFromUW-ExactAge-CertGender-time-status-prediction
,data=train_df)
```
```

7.2 Survival Tree approach code

```
```{r setup, include=FALSE}
library(ggplot2)
library(tidyverse)
library(survival)
library(rpart)
library(rpart.plot)
library(partykit)
library(rms)
library(dplyr)
library(infer)
library(Hmisc)
library(caret)

```
```

```
```{r}
# Input Tree
# Rank every node based on distance to the root omitting Gender and Age splits
# Prune those indices with ranking > 5

computePruneList <- function(frame, pruneLevel=5.0, discountedVariables = c("ExactAge", "CertG
  tol <- 0.000001
  vertexNumbers <- as.integer(row.names(frame))
  nverts <- length(vertexNumbers)
  newFrame <- frame
  newFrame$vertNums <- vertexNumbers
  newFrame <- newFrame[order(newFrame$vertNums),] # Order by vertex number

  vertexLevels <- rep(0, nverts)

  for (i in 2:nverts) {
    # Check if parent node split on age or gender
```
```

```
    vertexNumber <- newFrame$vertNums[i]
    parentNumber <- floor(vertexNumber/2.0)
    parentIndex <- match(parentNumber,newFrame$vertNums)
    discountedSplit <- (newFrame$var[parentIndex] %in% discountedVariables)

    # Update vertex levels
    if (!discountedSplit) {
      vertexLevels[i] <- vertexLevels[parentIndex] + 1
    }
  }
  #return(vertexLevels)
  #outputFrame <- cbind(newFrame$vertNums, vertexLevels)
  return(newFrame$vertNums[vertexLevels > pruneLevel + tol])
}

# Compute a vector of 5-year survival probabilities from an input vector of "surv" objects
computeSurvivalProbs <- function(survivalVector, year=5.0) {
  temp <- rep(0, length(survivalVector))
for (i in 1:length(survivalVector)) {
  index <- which.min(abs(survivalVector[[i]]$time - year))
  temp[i] <- survivalVector[[i]]$surv[index]
}
  return(temp)
}
```

```

```{r}
df <- read.csv("ITM2019BootcampDataCleaned.csv")


# Add pairwise questions
# for (i in 8:ncol(df)) {
#   for (j in 8:ncol(df)) {
#     df <- cbind(df, df[,i] | df[,j])
#     # We could easily check if adding interactions is beneficial
#   }
# }

test <- read.csv("ITM2019BootcampDataHoldoutNoOutcomes.csv")
set.seed(0)

# Code for allocating an internal test set
#ntrain <- floor(nrow(df)*0.9)
#ntest <- nrow(df) - ntrain
```
```

```
#train_ind <- sample(seq_len(nrow(df)), size = ntrain)
#train <- df[train_ind,]
#test <- df[-train_ind,]
train <- df
```

```{r}
# Survival Fit:
sfit = survfit(Surv(time=Duration, event = died) ~ 1, data=train)
plot(sfit)

costVector <- rep(1.0, ncol(df)-4)
costVector[1] <- 0.9
costVector[2] <- 0.9 # The default cost value is 1 for all variables

# Tree Fit:
tfit = rpart(formula = Surv(time=Duration, event = died) ~ . -X - Duration - died - YearsFromU

# Prune down to 5 questions, not counting age or gender splits

tfit.prune <- snip.rpart(tfit, toss = computePruneList(tfit$frame))

#rpart.plot(tfit.prune, uniform=TRUE, tweak=1.3)   # Plot tree after pruning down to 5-questic

plotcp(tfit.prune)
# Extract optimum value of regularization parameter
cpOpt <- tfit.prune$cptable[which.min(tfit.prune$cptable[,4]),1]

tfit.prune <- prune(tfit.prune, cp=cpOpt)
# Plot tree after split complexity pruning
rpart.plot(tfit.prune, uniform=TRUE, tweak=1.3)

```

```{r}
tfit2 <- as.party(tfit.prune)

# Extract Survival curves
predCurves <- rpart.predict(tfit2, newdata=test, type = "prob")

test$predicted5Year <- computeSurvivalProbs(predCurves)

plot(predCurves[[1]])  # Plot the estimated survival curve for person 1
#ggplot(test, aes(x=predicted5Year)) + geom_histogram()
```

```
write.csv(test$predicted5Year, file ="test5YearPredictions.csv", row.names=FALSE)

# Print out c-index on test set
#testFit <- Surv(time=test$Duration, event = test$died)
#rcorr.cens(x=test$predicted5Year, S=testFit)
# 0.73584 on test set
```

```
```