# Detecting Highly Predictive Pretreatment MicroRNA Signature For Cardiac Events In Lung Cancer Patients After Treatment

*5724*

*March 9, 2017*

```
#Data Preprocessing
names(data)[names(data)=="Framingham.Coronary.Heart.Disease.Risk.Score..MDCalc."] <- "F_score"
names(data)[names(data)=="Grade 2+ Cardiac Event"] <- "status"
names(data)[names(data)=="time.to.grade2"] <- "grade2_time"
names(data)[names(data)=="Concurrent.chemotherapy..0.no.1.yes."] <- "chemo"
names(data)[names(data)=="Volume Heart (cc)"] <- "volume"
names(data)[names(data)=="D0.5cc.LQ......2.5..EQD2Gy."] <- "max_dose.5"
names(data)[names(data)=="D2cc.LQ......2.5..EQD2Gy."] <- "max_dose2"


data <- mutate(data, id = rownames(data))


data$F_category <- 0
#Categorize Framinham Score based on risk
for (i in 1:nrow(data)) {
  if (data$F_score[i] == "NA"){
    data$F_category[i] = 3
  }
  else if (data$Gender.M.F[i] == "M") {
    if (as.numeric(data$F_score[i]) >= 17){
      data$F_category[i] = 2 #High risk
    }
    else if (as.numeric(data$F_score[i]) >= 12){
```

```
      data$F_category[i] = 1 #Medium risk
    }
    # else low risk
  } else{
    if (as.numeric(data$F_score[i]) >= 25){
      data$F_category[i] = 2
    }
    else if(as.numeric(data$F_score[i]) >= 20){
      data$F_category[i] = 1
    }
  }
}
```

```
data$SurvObj <- with(data, Surv(grade2_time,status))
## Kaplan-Meier estimator. The "log-log" confidence interval is preferred.
km.as.one <- survfit(SurvObj ~ 1, data = data, conf.type = "log-log")
#plot(km.as.one)
km.by.f <- survfit(SurvObj ~ F_category, data = data, conf.type = "log-log")
#km.by.f <- npsurv(SurvObj ~ F_category, data = data, conf.type = "log-log")
#survplot(km.by.f, main = 'Patients stratified by Framingham Category', xlab = 'Time (Months)', ylab =
```

```
fit <- coxph(Surv(grade2_time,status)~T.Stage+heart_Meandose+F_category,data)
sum_1 <- summary(fit)
c_index <- sum_1$concordance
stargazer(fit)
```

```
##
## % Table created by stargazer v.5.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvar
## % Date and time: Thu, Mar 09, 2017 - 01:04:36
## \begin{table}[!htbp] \centering
##   \caption{}
##   \label{}
## \begin{tabular}{@{\extracolsep{5pt}}lc}
```

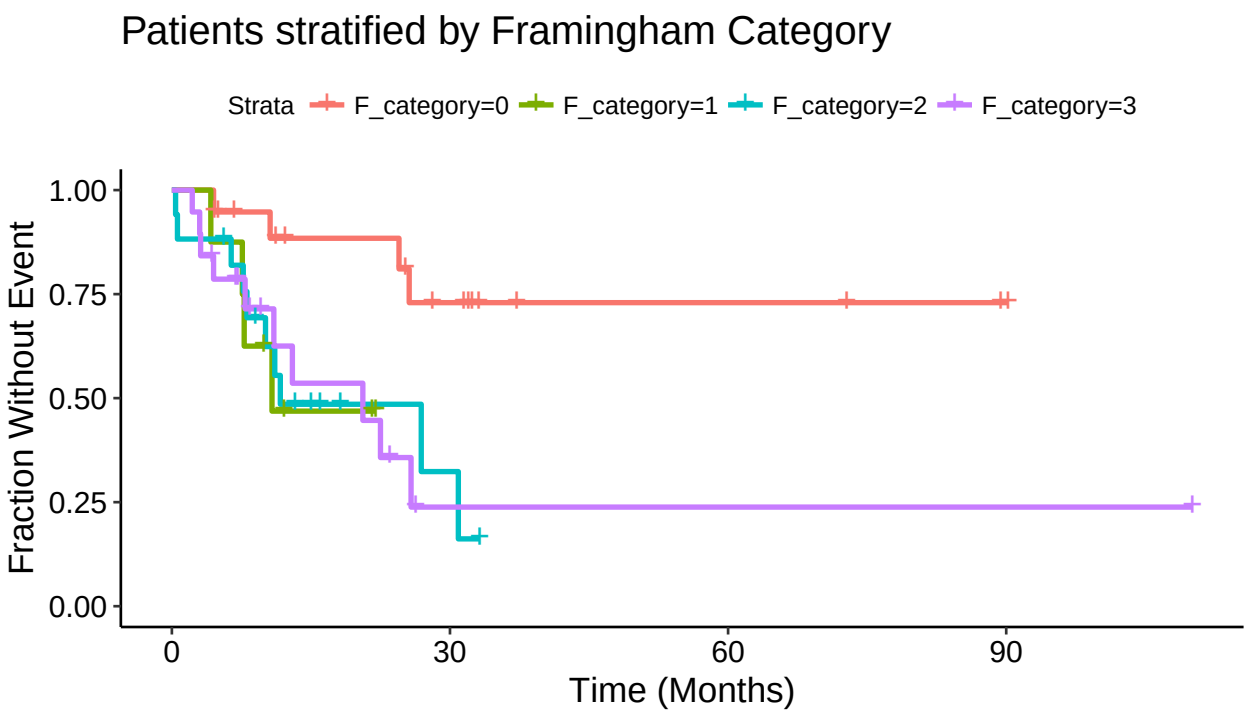



Figure 1: Kaplan-Meier Curves for three Framingham Categories

```
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
##  & \multicolumn{1}{c}{\textit{Dependent variable:}} \\
## \cline{2-2}
## \\[-1.8ex] & grade2\_time \\
## \hline \\[-1.8ex]
##  T.Stage & 0.335$^{*}$ \\
##   & (0.175) \\
##   & \\
##  heart\_Meandose & 0.055$^{**}$ \\
##   & (0.022) \\
##   & \\
##  F\_category & 0.465$^{**}$ \\
##   & (0.181) \\
##   & \\
## \hline \\[-1.8ex]
## Observations & 63 \\
```

```
## R$^{2}$ & 0.265 \\
## Max. Possible R$^{2}$ & 0.958 \\
## Log Likelihood & $-$89.818 \\
## Wald Test & 17.760$^{***}$ (df = 3) \\
## LR Test & 19.365$^{***}$ (df = 3) \\
## Score (Logrank) Test & 20.080$^{***}$ (df = 3) \\
## \hline
## \hline \\[-1.8ex]
## \textit{Note:}  & \multicolumn{1}{r}{$^{*}$p$<$0.1; $^{**}$p$<$0.05; $^{***}$p$<$0.01} \\
## \end{tabular}
## \end{table}
```

```
#Fit Cox PH Model without the microRNAs


#Diagonose the model by plotting residual plots
residual_plots <- function(fit){
  deviance <- resid(fit, type="deviance", collapse=data$id)
  pre <- predict(fit,type="risk",se.fit=TRUE)
  #plot(deviance,pre$fit) #Plot deviance residuals against fitted value
  plot(deviance,data$id, xlab = 'Deviance Residuals', ylab = 'Patients Id') #Plot deviance residuals ag
}
```

```
#Interpret the prediction results
accuracy <- function(fit){
  lp.pred <- predict(fit,type="lp",data=data)
  base <- basehaz(fit)
  pred_list <- list()
  target_list <- list()


  Pred.val <- base[32,1]*exp(lp.pred)
  Pred.target <- ifelse(Pred.val>median(Pred.val),1,0)
  target <- ifelse(data$grade2_time<=12,data$status,0)
```

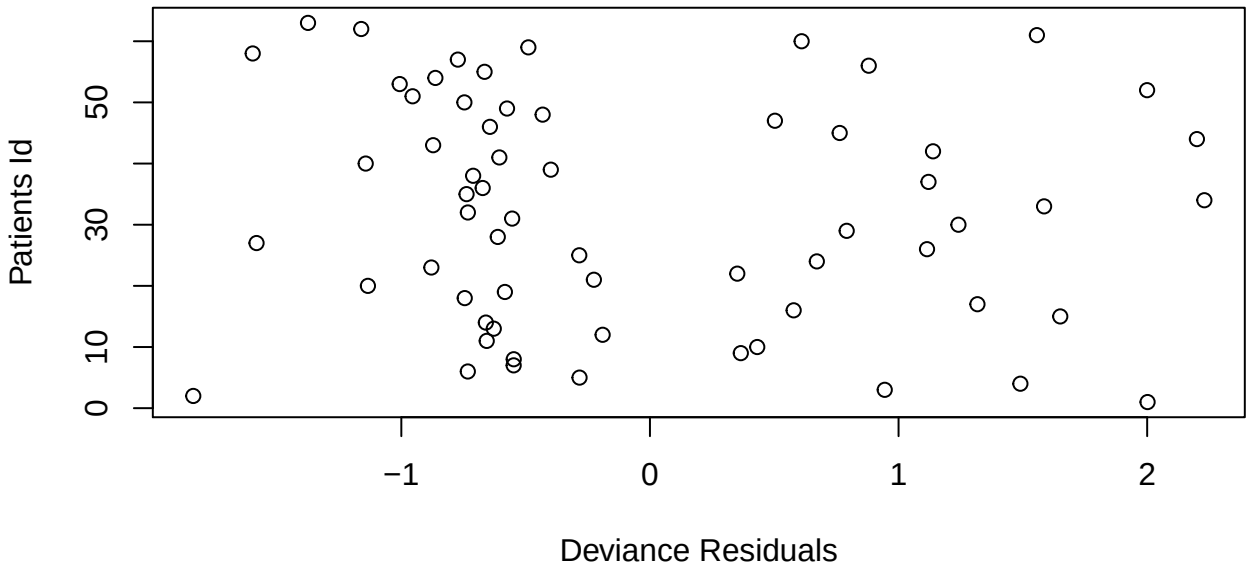
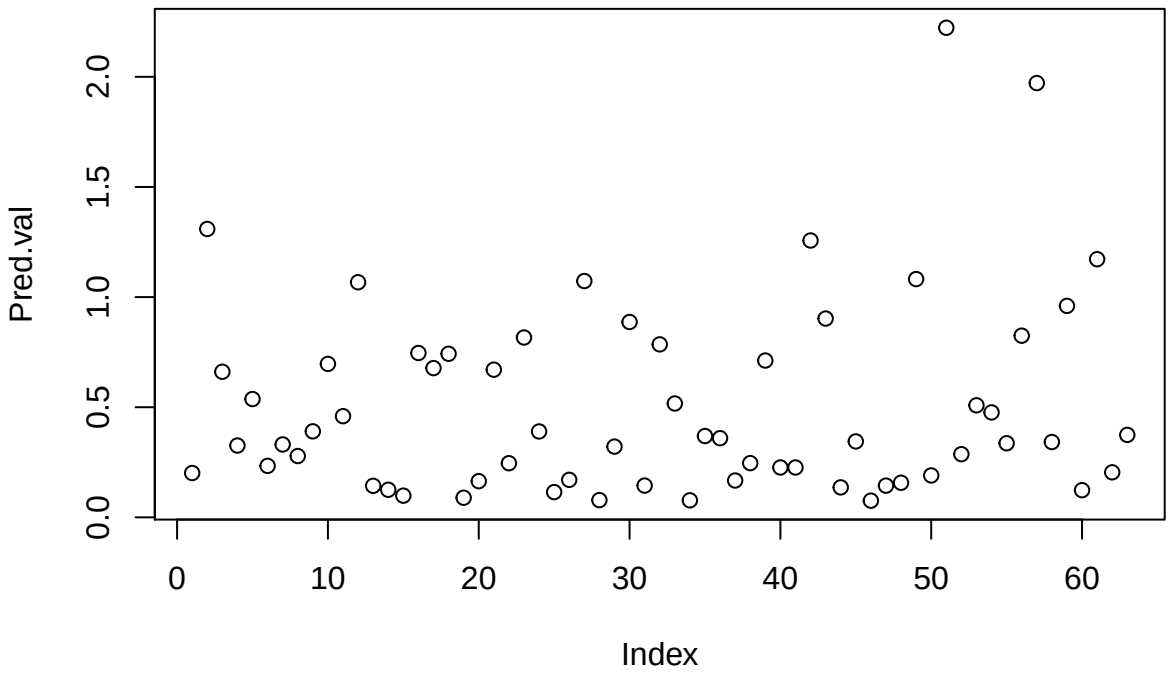



Figure 2: Deviance Residual Plot of Model 1

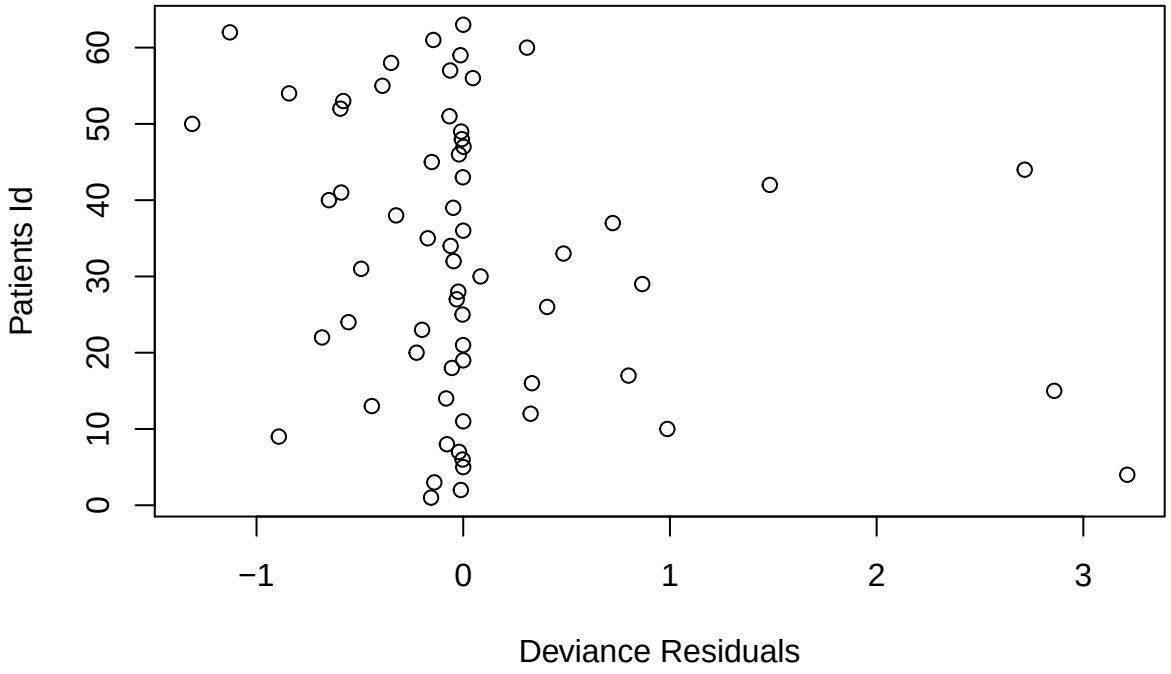
```
print(Pred.val)
print(summary(Pred.val))
plot(Pred.val)
print(Pred.target)
print(target)


#Create a confusion matrix
tp<-0
tn<-0
fp<-0
fn<-0
for (i in 1:length(target)){
  if (Pred.target[i] == 1 & target[i] == 1) tp <- tp + 1
  if (Pred.target[i] == 1 & target[i] == 0) fp <- fp + 1
  if (Pred.target[i] == 0 & target[i] == 1) fn <- fn + 1
  if (Pred.target[i] == 0 & target[i] == 0) tn <- tn + 1
}
percent <- function(x, digits = 2, format = "f", ...) {
  paste0(formatC(100 * x, format = format, digits = digits, ...), "%")
```

```
  }
  x <- c(tp/63,fp/63,fn/63,tn/63)
  confusion <- matrix(percent(x),ncol=2,byrow=TRUE)
  rownames(confusion) <- c("High Risk","Low Risk")
  colnames(confusion) <- c("Event","No Event")
  confusion <- as.table(confusion)
  print(confusion)
  accuracy = percent((tp+tn)/63)
  print(accuracy)

  x2 <- c(tp/(tp+fp),fp/(tp+fp),fn/(tn+fn),tn/(tn+fn))
  confusion2 <- matrix(percent(x2),ncol=2,byrow=TRUE)
  rownames(confusion2) <- c("High Risk","Low Risk")
  colnames(confusion2) <- c("Event","No Event")
  confusion2 <- as.table(confusion2)
  print(confusion2)
}
accuracy(fit)
```



```r
data2 <- data[,c(51,54,56:117)]
microRNA <- data[,c(56:117)]

#Fit Cox PH Model with subset of MicroRNAs
x <- model.matrix(~.-status-grade2_time,data2)
y <- Surv(data2$grade2_time,data2$status)

Lambdas <- function(...) {
  cv <- cv.glmnet(...)
  return(data.table(cvm=cv$cvm, lambda=cv$lambda))
}

OptimLambda <- function(k, ...) {
  require(parallel)
  require(data.table)
  MSEs <- data.table(rbind.fill(mclapply(seq(k), function(dummy) Lambdas(...))))
  return(MSEs[, list(mean.cvm=mean(cvm)), lambda][order(mean.cvm)][1]$lambda)
}

opt_lambda <- OptimLambda(k=100,x,y,family="cox",maxit=1000)

#We got 2 possible optimal lambdas: 0.2 and 0.031

fit2 <- glmnet(x,y,family="cox",maxit=1000)
#plot(fit2,label='T')
#plot(cv.fit2)
#Coefficients <- coef(fit2, s = 0.2)
Coefficients2 <- coef(fit2, s = 0.031)
#Coefficients <- coef(fit2, s = cv.fit2$lambda.1se)
Active.Index <- which(Coefficients2 != 0)
Active.Coefficients <- Coefficients2[Active.Index]

fit3 <- coxph(Surv(grade2_time,status)~T.Stage+heart_Meandose+F_category+NC08+NF10,data)
```

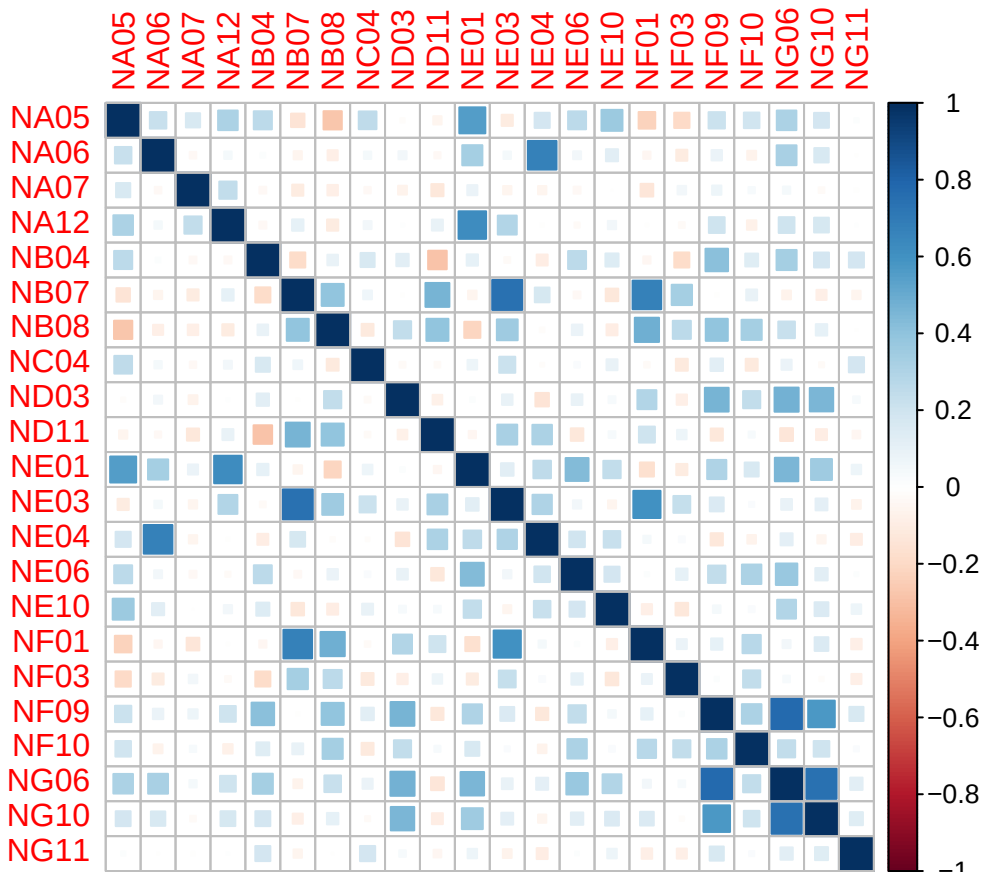
```
deviance3 <- resid(fit3, type="deviance", collapse=data$id)


fit4 <- coxph(Surv(grade2_time,status)~T.Stage+heart_Meandose+F_category+NA05+NA06+NA07+NA12+NB04+NB07+
residual_plots(fit4)
```




```
deviance4 <- resid(fit4, type="deviance", collapse=data$id)



vif_func<-function(in_frame,thresh=10,trace=T,...){

  #https://gist.github.com/fawda123/4717702

  require(fmsb)

  if(class(in_frame) != 'data.frame') in_frame<-data.frame(in_frame)

  #get initial vif value for all comparisons of variables
  vif_init<-NULL
  var_names <- names(in_frame)
```

```
for(val in var_names){
    regressors <- var_names[-which(var_names == val)]
    form <- paste(regressors, collapse = '+')
    form_in <- formula(paste(val, '~', form))
    vif_init<-rbind(vif_init, c(val, VIF(lm(form_in, data = in_frame, ...))))
    }
vif_max<-max(as.numeric(vif_init[,2]), na.rm = TRUE)

if(vif_max < thresh){
  if(trace==T){ #print output of each iteration
      prmatrix(vif_init,collab=c('var','vif'),rowlab=rep('',nrow(vif_init)),quote=F)
      cat('\n')
      cat(paste('All variables have VIF < ', thresh,', max VIF ',round(vif_max,2), sep=''),'\n\n')
      }
  return(var_names)
  }
else{

  in_dat<-in_frame

  #backwards selection of explanatory variables, stops when all VIF values are below 'thresh'
  while(vif_max >= thresh){

    vif_vals<-NULL
    var_names <- names(in_dat)

    for(val in var_names){
      regressors <- var_names[-which(var_names == val)]
      form <- paste(regressors, collapse = '+')
      form_in <- formula(paste(val, '~', form))
      vif_add<-VIF(lm(form_in, data = in_dat, ...))
      vif_vals<-rbind(vif_vals,c(val,vif_add))
```

```
        }
      max_row<-which(vif_vals[,2] == max(as.numeric(vif_vals[,2]), na.rm = TRUE))[1]

      vif_max<-as.numeric(vif_vals[max_row,2])

      if(vif_max<thresh) break

      if(trace==T){ #print output of each iteration
        prmatrix(vif_vals,collab=c('var','vif'),rowlab=rep('',nrow(vif_vals)),quote=F)
        cat('\n')
        cat('removed: ',vif_vals[max_row,1],vif_max,'\n\n')
        flush.console()
        }

      in_dat<-in_dat[,!names(in_dat) %in% vif_vals[max_row,1]]

      }

    return(names(in_dat))

    }

  }

#Multicollinearity check: since multicollinearity can reduce predictive power. If high bivariate correl
correlations <- cor(data2[Active.Index+1])
corrplot(correlations, method = "square")
```




```
vif_func(data2[Active.Index+1],thresh=10)


fit5 <- coxph(Surv(grade2_time,status)~T.Stage+heart_Meandose+F_category+NA05+ND11+NE01+NE03+NE10+NG06,d


sum_2 <- summary(fit3)
c_index_2 <- sum_2$concordance
```

```
data3 <- data2[Active.Index+1]
x2 <- model.matrix(~.,data3)
lambda_list <- c()
for (i in (1:10)){
  opt_lambda_2 <- OptimLambda(k=100,x2,y,family="cox",maxit=1000)
  lambda_list <- c(lambda_list,opt_lambda_2)
}
fit6 <- glmnet(x2,y,family="cox",maxit=1000)
```

```r
#Coefficients3 <- coef(fit6, s = 0.2)
Coefficients4 <- coef(fit6, s = 0.021)
Active.Index_2 <- which(Coefficients4 != 0)
Active.Coefficients_2 <- Coefficients4[Active.Index_2]
#NA06 and NE04 are removed
fit7 <- coxph(Surv(grade2_time,status)~T.Stage+heart_Meandose+F_category+NA05+NA07+NA12+NB04+NB07+NB08+
#Increased Accuracy and lowered BIC!
#Accuracy: 82.54%, BIC: 174.7493
```

```r
#Stepwise Variable Selection by AIC based on fit7
step <- stepAIC(fit7, direction="both")
```

```r
fit9 <- coxph(Surv(grade2_time,status)~T.Stage+heart_Meandose+F_category+NA05+NA07+NB04+NB07+NB08+ND03+
```

```r
stargazer(fit4,fit7,fit9)
```

```
##
## % Table created by stargazer v.5.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvar
## % Date and time: Thu, Mar 09, 2017 - 01:08:24
## \begin{table}[!htbp] \centering
##   \caption{}
##   \label{}
## \begin{tabular}{@{\extracolsep{5pt}}lccc}
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
##  & \multicolumn{3}{c}{\textit{Dependent variable:}} \\
## \cline{2-4}
## \\[-1.8ex] & \multicolumn{3}{c}{grade2\_time} \\
## \\[-1.8ex] & (1) & (2) & (3)\\
## \hline \\[-1.8ex]
##  T.Stage & 0.696 & 0.729$^{*}$ & 1.015$^{***}$ \\
##   & (0.423) & (0.423) & (0.383) \\
##   & & & \\
```

```
##  heart\_Meandose & 0.186$^{***}$ & 0.215$^{***}$ & 0.216$^{***}$ \\
##   & (0.062) & (0.067) & (0.060) \\
##   & & & \\
##  F\_category & 1.524$^{**}$ & 1.614$^{**}$ & 0.918$^{**}$ \\
##   & (0.684) & (0.705) & (0.443) \\
##   & & & \\
##  NA05 & 58,185.930$^{***}$ & 47,916.840$^{***}$ & 29,470.000$^{***}$ \\
##   & (19,365.470) & (17,115.030) & (11,228.770) \\
##   & & & \\
##  NA06 & 15.826 &  &  \\
##   & (60.874) &  &  \\
##   & & & \\
##  NA07 & $-$3,161.923 & $-$4,427.993$^{*}$ & $-$5,149.584 \\
##   & (2,250.015) & (2,410.608) & (3,542.100) \\
##   & & & \\
##  NA12 & $-$4,146.815 & $-$1,656.528 &  \\
##   & (2,680.481) & (1,923.785) &  \\
##   & & & \\
##  NB04 & 213.009$^{*}$ & 296.716$^{**}$ & 209.062$^{**}$ \\
##   & (122.809) & (120.915) & (102.777) \\
##   & & & \\
##  NB07 & 60,190.170 & 104,940.300$^{***}$ & 125,878.600$^{***}$ \\
##   & (43,139.050) & (39,047.100) & (35,654.690) \\
##   & & & \\
##  NB08 & $-$6,217.114 & $-$3,631.862 & $-$4,190.453$^{**}$ \\
##   & (3,805.790) & (2,245.350) & (1,706.836) \\
##   & & & \\
##  NC04 & $-$3,472.738 & $-$1,702.492 &  \\
##   & (2,318.406) & (1,920.903) &  \\
##   & & & \\
##  ND03 & $-$37.297 & $-$48.379$^{**}$ & $-$56.433$^{***}$ \\
##   & (25.251) & (22.277) & (21.759) \\
##   & & & \\
```

```
##  ND11 & 93,492.250$^{***}$ & 55,310.390$^{**}$ & 34,880.380$^{*}$ \\
##   & (36,041.460) & (27,044.480) & (20,604.720) \\
##   & & & \\
##  NE01 & $-$659.189$^{**}$ & $-$559.481$^{**}$ & $-$724.050$^{***}$ \\
##   & (306.931) & (263.452) & (205.155) \\
##   & & & \\
##  NE03 & 15,581.560$^{***}$ & 10,242.050$^{**}$ & 7,507.496$^{***}$ \\
##   & (5,752.479) & (4,660.445) & (2,831.056) \\
##   & & & \\
##  NE04 & $-$5,652.448 &  &  \\
##   & (4,304.019) &  &  \\
##   & & & \\
##  NE06 & $-$25.625 & $-$43.726 & $-$50.262$^{**}$ \\
##   & (30.684) & (26.792) & (24.563) \\
##   & & & \\
##  NE10 & $-$63,035.040$^{***}$ & $-$52,011.940$^{***}$ & $-$36,670.410$^{***}$ \\
##   & (19,008.340) & (17,583.470) & (13,226.710) \\
##   & & & \\
##  NF01 & 314.713 & $-$2,539.117 & $-$5,625.556 \\
##   & (5,982.087) & (5,084.110) & (4,150.676) \\
##   & & & \\
##  NF03 & $-$8,875.256$^{*}$ & $-$12,806.920$^{***}$ & $-$14,274.530$^{***}$ \\
##   & (4,535.732) & (4,760.396) & (4,181.286) \\
##   & & & \\
##  NF09 & $-$332.954 & $-$569.849 & $-$513.724$^{*}$ \\
##   & (434.701) & (401.003) & (300.409) \\
##   & & & \\
##  NF10 & $-$564.911 & $-$72.117 &  \\
##   & (799.021) & (460.446) &  \\
##   & & & \\
##  NG06 & 2,110.960$^{***}$ & 1,582.562$^{***}$ & 1,698.116$^{***}$ \\
##   & (616.950) & (445.635) & (408.810) \\
##   & & & \\
```

```
##  NG10 & $-$321.544 & $-$46.423 &  \\
##   & (275.246) & (139.314) &  \\
##   & & & \\
##  NG11 & 63.694$^{*}$ & 57.884$^{*}$ &  \\
##   & (34.168) & (33.175) &  \\
##   & & & \\
## \hline \\[-1.8ex]
## Observations & 63 & 63 & 63 \\
## R$^{2}$ & 0.801 & 0.790 & 0.780 \\
## Max. Possible R$^{2}$ & 0.958 & 0.958 & 0.958 \\
## Log Likelihood & $-$48.668 & $-$50.358 & $-$51.814 \\
## Wald Test & 29.980 (df = 25) & 29.310 (df = 23) & 29.020$^{**}$ (df = 18) \\
## LR Test & 101.665$^{***}$ (df = 25) & 98.285$^{***}$ (df = 23) & 95.373$^{***}$ (df = 18) \\
## Score (Logrank) Test & 62.979$^{***}$ (df = 25) & 62.534$^{***}$ (df = 23) & 50.881$^{***}$ (df = 18
## \hline
## \hline \\[-1.8ex]
## \textit{Note:}  & \multicolumn{3}{r}{$^{*}$p$<$0.1; $^{**}$p$<$0.05; $^{***}$p$<$0.01} \\
## \end{tabular}
## \end{table}
```

```
lp.pred <- predict(fit9,type="lp",data=data)
base <- basehaz(fit9)


sig <- exp(lp.pred)
data$Pred.target <- ifelse(sig>quantile(sig,0.75),3,ifelse(sig>median(sig),2,ifelse(sig>quantile(sig,0.1
data$SurvObj <- with(data, Surv(grade2_time,status))
km.by.p <- survfit(SurvObj ~ Pred.target, data = data, conf.type = "log-log")
```

```
lp.pred_1 <- predict(fit,type="lp",data=data)
base <- basehaz(fit)
val1 <- exp(lp.pred_1)
data$Pred.target_1 <- ifelse(val1>quantile(val1,0.75),3,ifelse(val1>median(val1),2,ifelse(val1>quantile
data$SurvObj <- with(data, Surv(grade2_time,status))
```

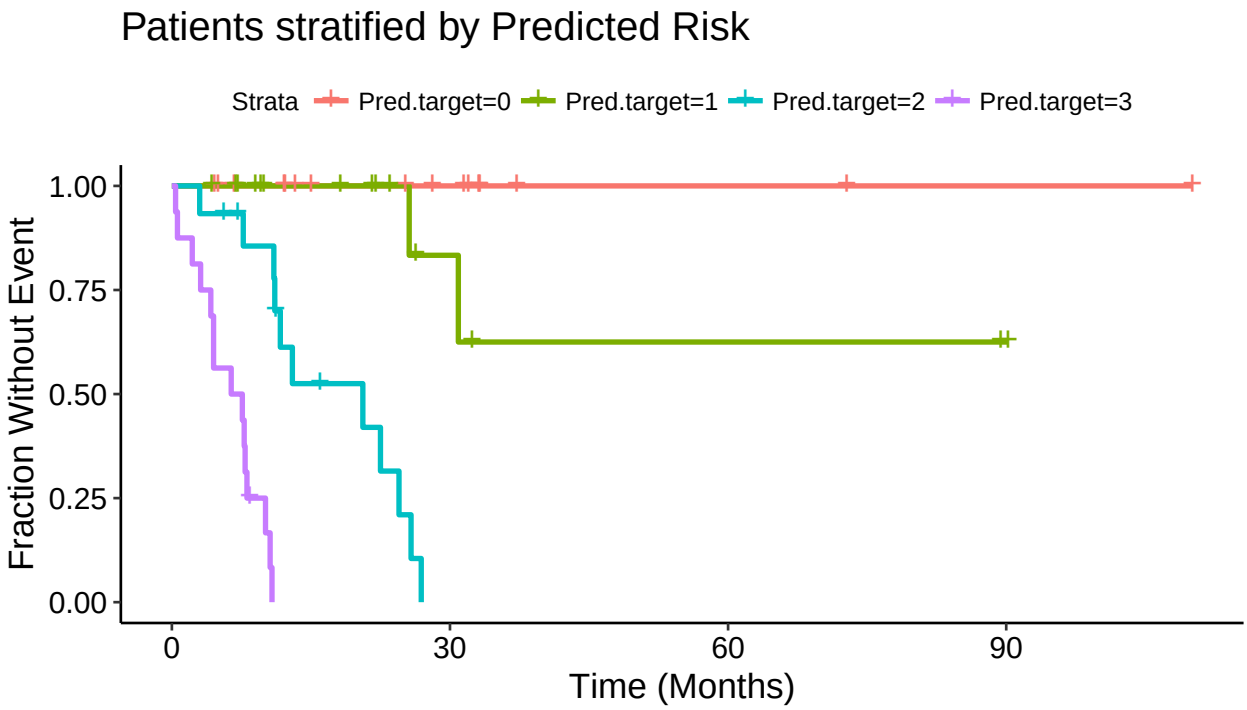



Figure 3: K-M Plot of Model 4

```
km.by.p_1 <- survfit(SurvObj ~ Pred.target_1, data = data, conf.type = "log-log")
```

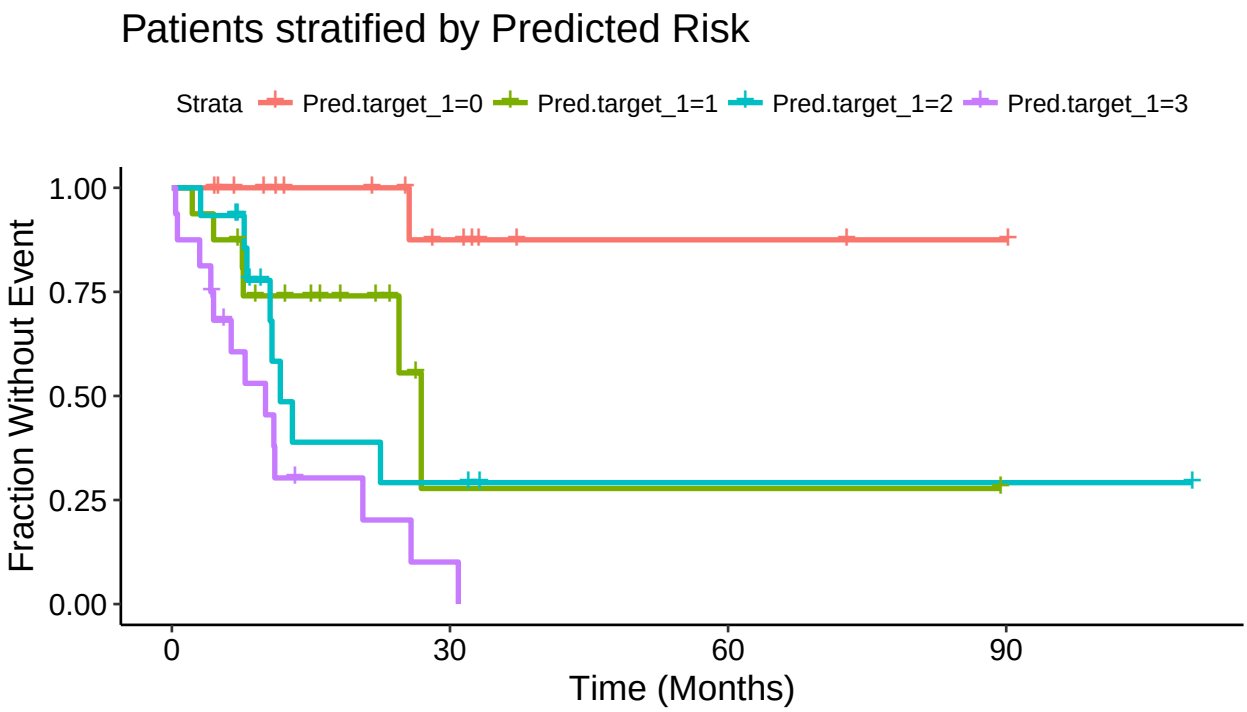



Figure 4: K-M Plot of Model 1