Table of Contents

Introduction & Background

Diagnostic and prognostic models are typically evaluated with measures of accuracy that do not address clinical consequences. Decision-analytic techniques allow assessment of clinical outcomes but often require collection of additional information and may be cumbersome to apply to models that yield a continuous result. Decision curve analysis is a method for evaluating and comparing prediction models that incorporates clinical consequences, requires only the data set on which the models are tested, and can be applied to models that have either continuous or dichotomous results. This document will walk you through how to perform a decision curve analysis (DCA) in many settings, and how to interpret the resulting curves. In DCA prediction models are compared to two default strategies: 1) assume that all patients are test positive and therefore treat everyone, or 2) assume that all patients are test negative and offer treatment to no one. “Treatment” is considered in the widest possible sense, not only drugs, radiotherapy or surgery, but advice, further diagnostic procedures or more intensive monitoring. For more details on DCA, visit decisioncurveanalysis.org. You’ll find the original articles explaining the details of the DCA derivation along with other papers providing more details.

Below we’ll walk you through how to perform DCA for binary and time-to-event outcomes. Also included is the Stata, R and SAS code to perform DCA. While the code is available in multiple programming languages, the figures in this document were all created using Stata 12.0.

Note to Stata users: The curves present in this document may not look the same as the curves you get when running the code. The following options have been added to each DCA for formatting purposes:

```
legend(size(vsmall) cols(1) textwidth(100)) scheme(s1color) ylabel(, format("%9.2f"))
xlabel(, format("%9.2f"))
```

Decision Curve Analysis for Binary Outcomes

Motivating Example

We'll be working with the example dataset (dca.txt, dca.dta, dca.R, dca.sas7bdat), available on our website. The dataset includes information on 750 patients who have recently discovered they have a gene mutation that puts them at a higher risk for harboring cancer. Each patient has been biopsied and we know their cancer status. It is known that older patients with a family history of cancer have a higher probability of harboring cancer. A clinical chemist has recently discovered a marker that she believes can distinguish between patients with and without cancer. We wish to assess whether or not the new marker does indeed identify patients with and without cancer. If the marker does indeed predict well, many patients will not need to undergo a painful biopsy.

The example dataset contains the following information:

Variable	**Description**
patientid	Identification Number
cancer	Cancer Diagnosis: 0=No, 1=Yes
risk_group	Patient Risk Group (Low, Intermediate, High)
age	Patient Age, years
famhistory	Family History of Cancer: 0=No, 1=Yes
marker	Marker
cancerpredmarker	Prob. of Cancer based on Age, Family History, and Marker

Basic Data Set-up

Here we will go through step by step how to import your data, build models based on multiple variables, and use those models to obtain predicted probabilities. The code below assumes you have your data and DCA function code files saved in a folder called "C:\Decision Curve Analysis". The first step is to import your data.

Stata Code

```
//Set our directory
cd "C:\Decision Curve Analysis"
use "dca.dta", clear
```

R Code

```
#Set our directory
setwd("C:\\Decision Curve Analysis")
#Source file to use dca command
source("dca.R")
data.set = read.delim("dca.txt", header=TRUE, sep="\t")
attach(data.set)
```

SAS Code

```
*Set our directory;
LIBNAME HOME "C:\Decision Curve Analysis";
*Source file to use dca command
OPTIONS MAUTOSOURCE SASAUTOS=(sasautos home);
DATA dca; SET home.origdca;
RUN;
```

Once you have the data imported, you'll want to begin building your model. As we have a binary outcome (i.e. the outcome of our model has two levels: cancer or no cancer), we will be using a logistic regression model.

Univariate Decision Curve Analysis

First, we want to confirm family history of cancer is indeed associated with the biopsy result. Via logistic regression with cancer as the outcome, we can see that family history is related to biopsy outcome (OR 1.80; 95% CI 1.09, 2.99; p=0.022). The DCA can help us address the clinical utility of using family history to predict biopsy outcome.

Stata Code

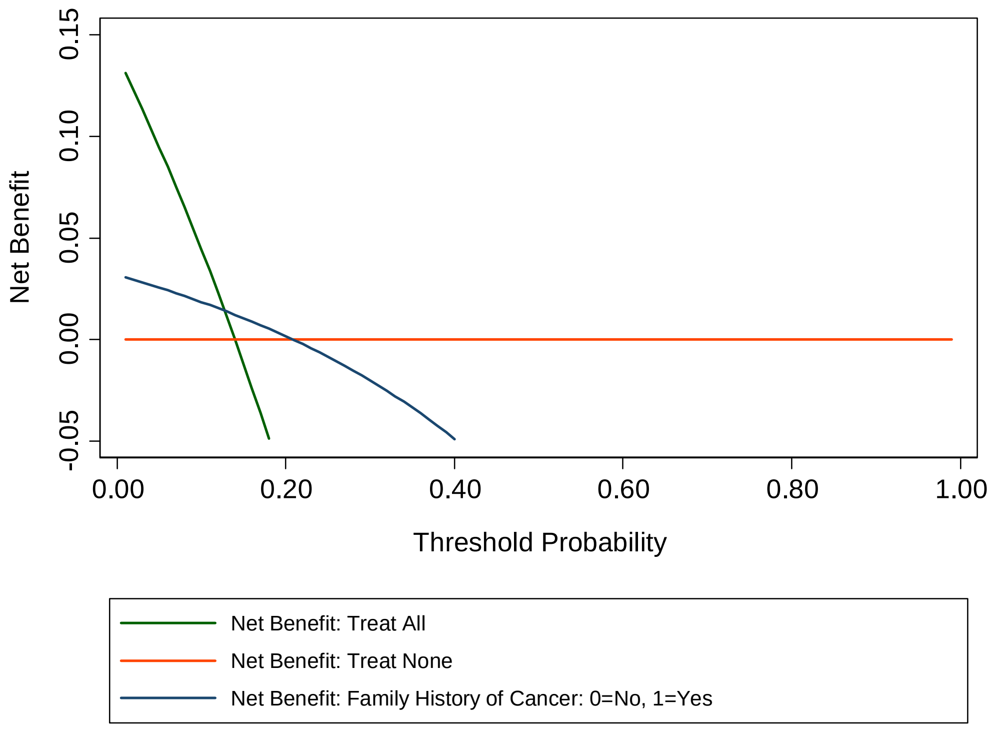
```
//Test whether family history is associated with cancer
logit cancer famhistory
//Run the decision curve: family history is coded as 0 or 1, i.e. a probability
//so no need to specify the "probability" option
dca cancer famhistory
```

R Code

```
#Test whether family history is associated with cancer
summary(glm(cancer ~ famhistory, family=binomial(link="logit")))
#Run the decision curve: family history is coded as 0 or 1, i.e. a probability
#so no need to specify the "probability" option
dca(data=data.set, outcome="cancer", predictors="famhistory")
```

SAS Code

```
*Test whether family history is associated with cancer;
PROC LOGISTIC DATA=dca DESCENDING;
       MODEL cancer = famhistory;
RUN;
*Run the decision curve: family history is coded as 0 or 1, i.e. a probability, so
no need to specify the "probability" option;
%DCA(data=dca, outcome=cancer, predictors=famhistory, graph=yes);
```



First, note that there are many threshold probabilities shown here that are not of interest. For example, it is unlikely that a patient would demand that they had at least a 50% risk of cancer before they would accept a biopsy. Let's do the DCA again, this time restricting the output to threshold probabilities a more clinically reasonable range, between 0% and 35% with the `xstop` option.

Stata Code

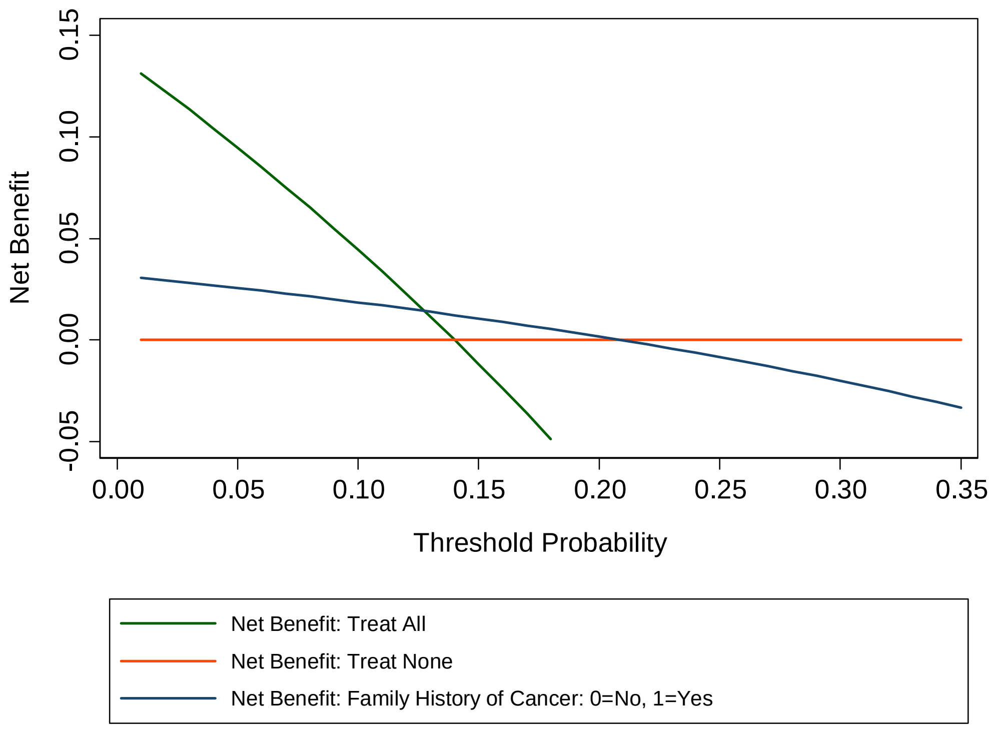
```
dca cancer famhistory, xstop(0.35) xlabel(0(0.05)0.35)
```

R Code

```
dca(data=data.set, outcome="cancer", predictors="famhistory", xstop=0.35)
```

SAS Code

```
%DCA(data=dca, outcome=cancer, predictors=famhistory, graph=yes, xstop=0.35);
```



Now that the graph is showing a more reasonable range of threshold probabilities, let's assess the clinical utility of family history alone. We can see here that although family history is significantly associated with biopsy outcome, it only adds value to a small range of threshold probabilities near 13% - 20%. If your personal threshold probability is 15% (i.e. you would undergo a biopsy if your probability of cancer was greater than 15%), then family history alone can be beneficial in making the decision to undergo biopsy. However, if your threshold probability is less than 13% or higher than 20%, then family history adds no more benefit than a biopsy all, or biopsy none scheme.

Multivariable Decision Curve Analysis

Evaluation of New Models

We wanted to examine the value of a statistical model that incorporates family history, age, and the marker. First we will build the logistic regression model with all three variables, and second we would have saved out the predicted probability of having cancer based on the model. Note that in our example dataset, this variable actually already exists so it wouldn't be necessary to create the predicted probabilities once again.

Stata Code

```
//run the multivariable model
logit cancer marker age famhistory
//save out predictions in the form of probabilities
predict cancerpredmarker
```

R Code

```
#run the multivariable model
model = glm(cancer ~ marker + age + famhistory, family=binomial(link="logit"))
#save out predictions in the form of probabilities
data.set$cancerpredmarker = predict(model, type="response")
```

SAS Code

```
*run the multivariable model;
PROC LOGISTIC DATA=dca DESCENDING;
      MODEL cancer = marker age famhistory;
      *save out predictions in the form of probabilities;
      SCORE CLM OUT=dca (RENAME=(P_1=cancerpredmarker));
RUN;
```

We now want to compare our different approaches to cancer detection: biopsying everyone, biopsying no-one, biopsying on the basis of family history, or biopsying on the basis of a multivariable statistical model including the marker, age and family history of cancer.

Stata Code

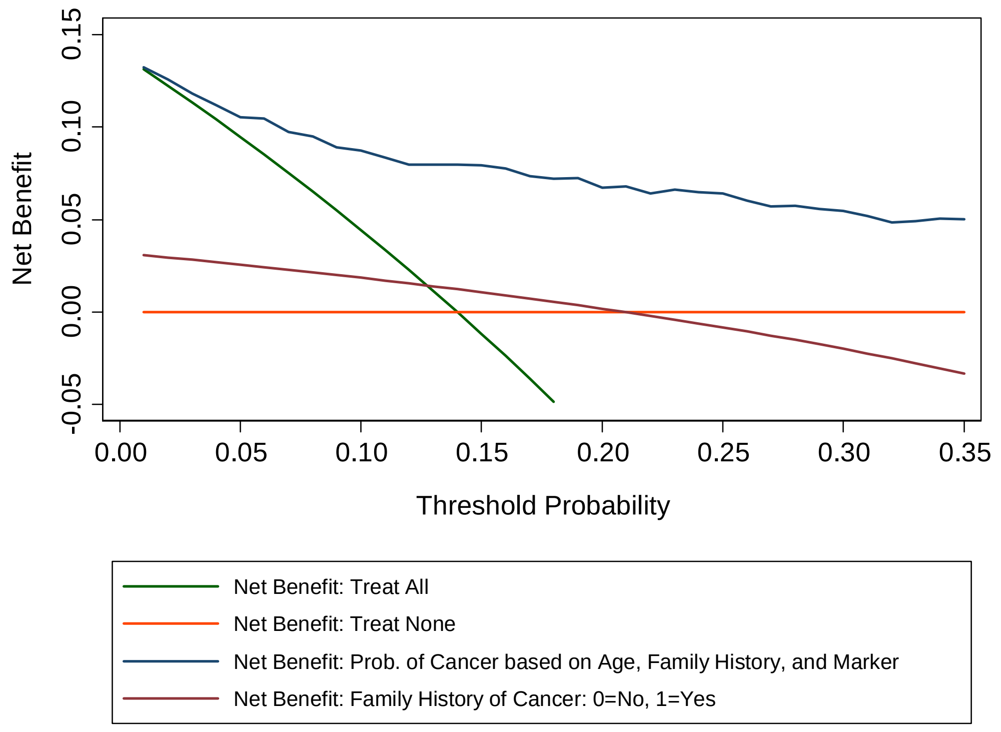
```
dca cancer cancerpredmarker famhistory, xstop(0.35) xlabel(0(0.05)0.35)
```

R Code

```
dca(data=data.set, outcome="cancer", predictors=c("cancerpredmarker","famhistory"),
      xstop=0.35)
```

SAS Code

```
%DCA(data=dca, outcome=cancer, predictors=cancerpredmarker famhistory, graph=yes,
xstop=0.35);
```



The key aspect of decision curve analysis is to look at which strategy leads to the largest net benefit (i.e. the “highest” line), which in this example would correspond to the model that includes age, family history of cancer, and the marker. It is clear that, across the range of reasonable threshold probabilities, one cannot go wrong basing decisions on this multivariate model: it is superior, and unlike any alternative strategy, it is never worse.

A few points are worth noting. First, look at the green line, the net benefit for “treat all”, that is, biopsy everyone. This crosses the y axis at the prevalence. Imagine that a man had a risk threshold of 14%, and asked his risk under the "biopsy all" strategy. He would be told that his risk was the prevalence (14%). When a patient’s risk threshold is the same as his predicted risk, the net benefit of biopsying and not biopsying are the same.

Second, the decision curve for the binary variable (family history of cancer, the brown line) crosses the “biopsy all men” line at 1 – negative predictive value and again, this is easily explained: the negative predictive value is 87%, so a patient with no family history of cancer has a probability of disease of 13%; a patient with a threshold probability less than this – for example, a patient who would opt for biopsy even if risk was 10% - should therefore be biopsied even if he/she had no family history of cancer. The decision curve for a binary variable is equivalent to biopsy no-one at the positive predictive value. This is because for a binary variable, a patient with the characteristic is given a risk at the positive predictive value.

Evaluation of Published Models

Imagine that a model was published by Brown et al. with respect to our cancer biopsy data set. The authors reported a statistical model with coefficients of 0.75 for a positive family history of cancer; 0.26 for each increased year of age, and an intercept of -17.5. To test this formula on our dataset:

Stata Code

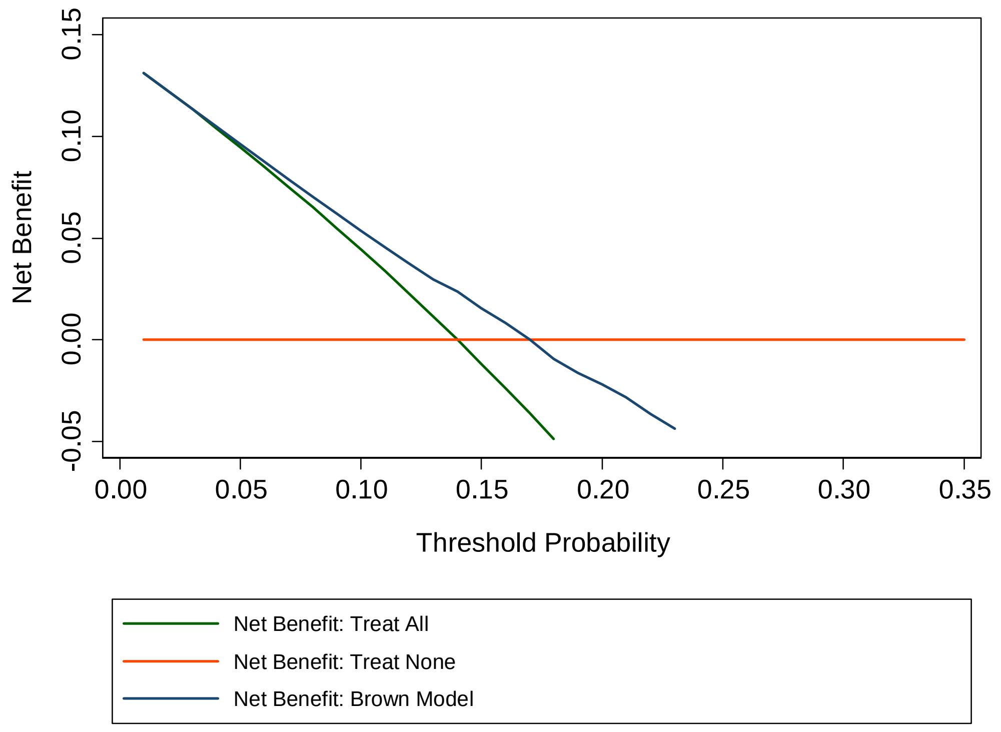
```
//Use the coefficients from the Brown model
g logodds_Brown = 0.75*(famhistory) + 0.26*(age) - 17.5
//Convert to predicted probability
g phat_Brown = invlogit(logodds_Brown)
label var phat_Brown "Risk from Brown Model"
//Run the decision curve
dca cancer phat_Brown, xstop(0.35) xlabel(0(0.05)0.35)
```

R Code

```
#Use the coefficients from the Brown model
logodds_Brown = 0.75*(famhistory)+0.26*(age)-17.5
#Convert to predicted probability
data.set$phat_Brown = exp(logodds_Brown)/(1+exp(logodds_Brown))
#Run the decision curve
dca(data=data.set, outcome="cancer", predictors="phat_Brown", xstop=0.35)
```

SAS Code

```
DATA dca; SET dca;
       *use the coefficients from the Brown model;
       logodds_Brown = 0.75*(famhistory) + 0.26*(age) - 17.5;
       *convert to predicted probability;
       phat_Brown = exp(logodds_Brown) / (1 + exp(logodds_Brown));
RUN;
*run the decision curve;
%DCA(data=dca, outcome=cancer, predictors=phat_Brown, xstop=0.35);
```



This decision curve suggests that although the model might be useful in the most risk averse patients, it is actually harmful in patients with more moderate threshold probabilities. As such, the Brown et al. model should not be used in clinical practice. This effect, a model being harmful, occurs due to miscalibration, that is, when patients are given risks that are too high or too low. Note that miscalibration only occurs rarely when models are created and tested on the same data set, such as in the example where we created a model with both family history and the marker.

Joint or Conditional Tests

Many decisions in medicine are based on joint or conditional test results. A classic example is where patients are categorized on the basis of a test as being at high, low or intermediate risk. Patients at high risk are referred immediately for treatment (in our example biopsied); patients at low risk are reassured and advised that no further action is necessary; patients at intermediate risk are sent for an additional test, with subsequent treatment decisions made accordingly.

Imagine that for our example there was a previous test that categorized our patients as high, low, and intermediate risk of cancer and we wanted to incorporate our marker. There are five clinical options:

1. Biopsy everyone
2. Biopsy no-one
3. Biopsy everyone that was determined to be at high risk of cancer; don't use the marker
4. Measure the marker for everyone, then biopsy anyone who is *either* at high risk of cancer *or* who was determined to have a probability of cancer past a certain level, based on the marker (i.e. joint approach)

5. Biopsy everyone at high risk; measure the marker for patients at intermediate risk and biopsy those with a probability of cancer past a certain level, based on the marker (i.e. conditional approach)

Decision curve analysis can incorporate joint or conditional testing. All that is required is that appropriate variables are calculated from the data set; decision curves are then calculated as normal. First we would create the variables to represent our joint and conditional approach. For our example, let us use 0.15 as the cutoff probability level for patients who had their marker measured and should be biopsied.

Stata Code

```
//Create a variable for the strategy of treating only high risk patients
//This will be 1 for treat and 0 for don’t treat
g high_risk = risk_group=="high"
label var high_risk "Treat Only High Risk Group"
//Treat based on Joint Approach
g joint = risk_group =="high" | cancerpredmarker > 0.15
label var joint "Treat via Joint Approach"
//Treat based on Conditional Approach
g conditional = risk_group=="high" | ///
       (risk_group == "intermediate" & cancerpredmarker > 0.15)
label var conditional "Treat via Conditional Approach"
```

R Code

```
#Create a variable for the strategy of treating only high risk patients
#This will be 1 for treat and 0 for don’t treat
data.set$high_risk = ifelse(risk_group=="high", 1, 0)
#Treat based on Joint Approach
data.set$joint = ifelse(risk_group=="high" | cancerpredmarker > 0.15, 1, 0)
# Treat based on Conditional Approach
data.set$conditional = ifelse(risk_group=="high" | (risk_group=="intermediate" &
       cancerpredmarker > 0.15), 1, 0)
```

SAS Code

```
*Create a variable for the strategy of treating only high risk patients;
*This will be 1 for treat and 0 for don't treat;
DATA dca;
       SET dca;
       high_risk = (risk_group="high");
       LABEL high_risk="Treat Only High Risk Group";
       *Treat based on joint approach;
       joint = (risk_group="high") | (cancerpredmarker > 0.15);
       LABEL joint="Treat via Joint Approach";
       *Treat based on conditional approach;
       conditional = (risk_group="high") |
              (risk_group="intermediate" & cancerpredmarker > 0.15);
       LABEL conditional="Treat via Conditional Approach";
RUN;
```

Now that we have the variables worked out, we can run the decision curve analysis.

Stata Code

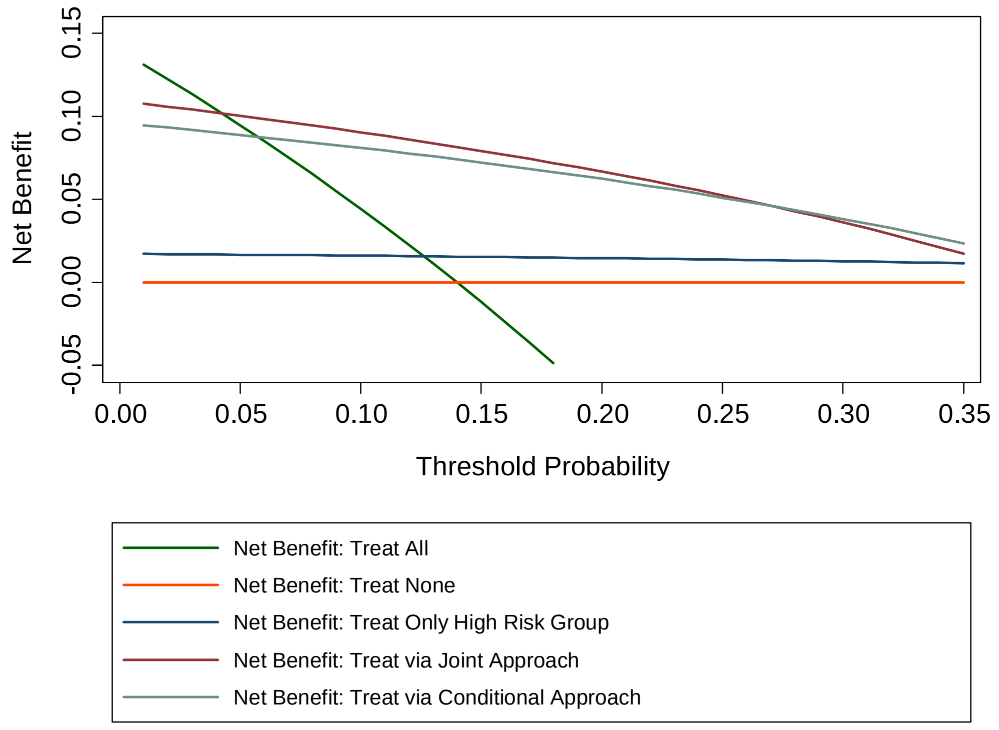
```
dca cancer high_risk joint conditional, xstop(0.35)
```

R Code

```
dca(data=data.set, outcome="cancer", predictors=c("high_risk", "joint",
       "conditional"), xstop=0.35)
```

SAS Code

```
%DCA(data=dca, outcome=cancer, predictors=high_risk joint conditional, graph=yes,
xstop=0.35);
```



This appears to show that the joint test is the better option for the range of threshold probabilities from 5% - 24% since it has the highest net benefit across that range. Less than 5%, the clinical option of treating all would be superior to any other option, though rarely would treatment thresholds be so low. From 28%-35%, the conditional test would be a slightly better option, and in between the two ranges, the joint and conditional tests are comparable. The obvious disadvantage of the joint test is that the marker needs to be measured for everyone, and such tests may be expensive and time consuming.

Incorporating Harms into Model Assessment

To incorporate the harm of testing and measuring the marker, we ask a clinician, who tells us that, even if the marker were perfectly accurate, few clinicians would conduct more than 30 tests to predict one cancer diagnosis. This might be because the test is expensive, or requires some invasive procedure to obtain. The “harm” of measuring the marker is the reciprocal of 30, or 0.0333.

To construct the decision curves for each strategy we now incorporate harm. We have to calculate harm specially for the conditional test, because only patients at intermediate risk are measured for the marker. Then incorporate it into our decision curve. The strategy for incorporating harm for the conditional test is by multiplying the proportion scanned by the harm of the scan.

Stata Code

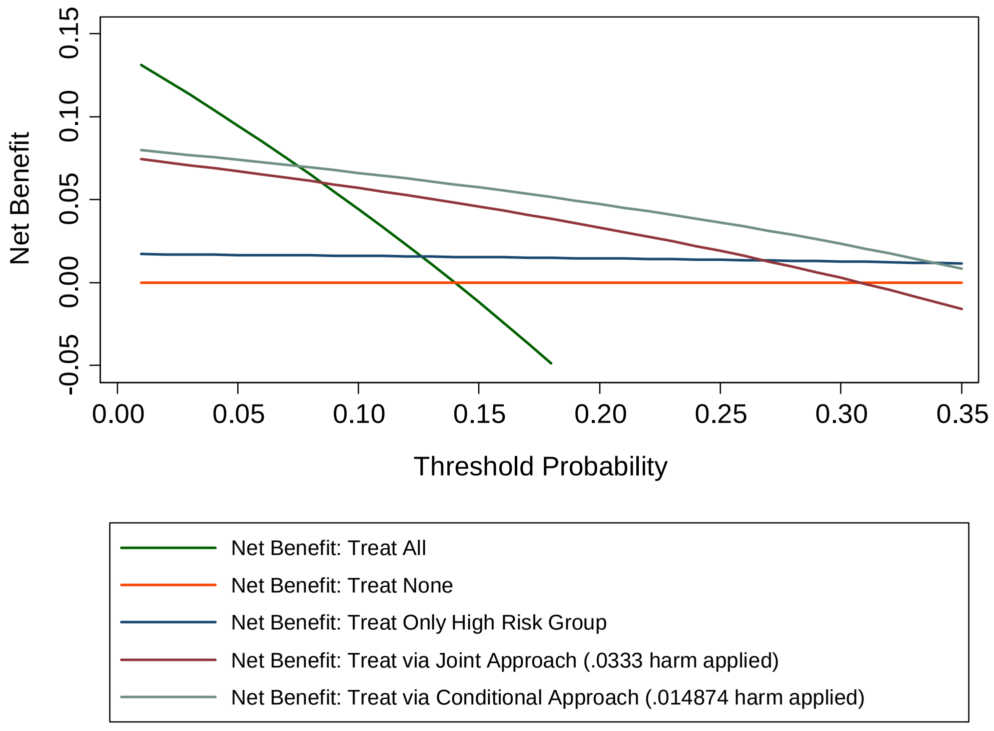
```
//the harm of measuring the marker is stored in a local
local harm_marker = 0.0333
//in the conditional test, only patients at intermediate risk
//have their marker measured
g intermediate_risk = (risk_group=="intermediate")
//harm of the conditional approach is proportion of patients who have the marker
//measured multiplied by the harm
sum intermediate_risk
local harm_conditional = r(mean)*`harm_marker'
//Run the decision curve
dca cancer high_risk joint conditional, ///
    harm(0 `harm_marker' `harm_conditional') xstop(0.35) xlabel(0(0.05)0.35)
```

R Code

```
#the harm of measuring the marker is stored in a scalar
harm_marker = 0.0333
#in the conditional test, only patients at intermediate risk
#have their marker measured
intermediate_risk = ifelse(risk_group=="intermediate", c(1), c(0))
#harm of the conditional approach is proportion of patients who have the marker
#measured multiplied by the harm
harm_conditional = mean(intermediate_risk)*harm_marker
#Run the decision curve
dca(data=data.set, outcome="cancer", predictors=c("high_risk", "joint",
       "conditional"), harm=c(0, harm_marker, harm_conditional),
       xstop=0.35)
```

SAS Code

```
*the harm of measuring the marker is stored as a macro variable;
%LET harm_marker = 0.0333;
*in the conditional test, only patients at intermediate risk have their marker
measured;
DATA dca; SET dca;
       intermediate_risk = (risk_group="intermediate");
RUN;
*calculate the proportion of patients who have the marker and save out mean risk;
PROC MEANS DATA=dca;
       VAR intermediate_risk;
       OUTPUT OUT=meanrisk MEAN=meanrisk;
RUN;
DATA _NULL_; SET meanrisk;
       CALL SYMPUT("meanrisk",meanrisk);
RUN;
*harm of the conditional approach is proportion of patients who have the marker
measured multiplied by the harm;
%LET harm_conditional = %SYSEVALF(&meanrisk.*&harm_marker.);
*Run the decision curve;
%DCA(data=dca, outcome=cancer, predictors=high_risk joint conditional, harm=0
&harm_marker. &harm_conditional., xstop=0.35);
```



Here the conditional test is clearly the best option (above the 8% treatment threshold), in fact, once you take into account the harm of measuring the marker, it is clearly not worth measuring the marker for everyone: the net benefit of just treating high risk patients is often higher than that of the joint test.

Saving out Net Benefit Values

For any model assessed through decision curve analysis, if we also wanted to show the net benefits in a table, we could save them out. We would simply need to specify the name of the file we'd like it to be saved as. For a particular range of value, we would only need to specify what threshold to start, stop, and the increment we'd like to use. To assess or display the increase in net benefit we'd simply need to subtract the net benefit of the model based on treating all patients from the net benefit of our model.

Let us imagine that we want to view the net benefit of using only the marker to predict whether a patient has cancer, compared with the net benefits of biopsying all patients at thresholds of 5%, 10%, 15% … 35%.

For the model itself, we would actually need to first specify that the marker variable – unlike those of any of the models before – is not a probability. Based on our thresholds, we'd want to begin at 0.05, and by increments of 0.05, stop at 0.35. As we are not interested in the graph, we can also specify to suppress the graph from being displayed.

Stata Code

```
//Run the decision curve and save out net benefit results
//Specifying xby(.05) since we'd want 5% increments
dca cancer marker, prob(no) xstart(0.05) xstop(0.35) xby(0.05) nograph ///
       saving("DCA Output marker.dta", replace)
//Load the data set with the net benefit results
use "DCA Output marker.dta", clear
//Calculate difference between marker and treat all
//Our standard approach is to biopsy everyone so this tells us
//how much better we do with the marker
g advantage = marker – all
label var advantage "Increase in net benefit from using Marker model"
```

R Code

```
#Run the decision curve, specify xby=0.05 since we want 5% increments
output = dca(data=data.set, outcome="cancer", predictors="marker", probability=F,
       xstart=0.05, xstop=0.35, xby=0.05, graph=F)
#Calculate difference between marker and treat all
#Our standard approach is to biopsy everyone so this tells us
#how much better we do with the marker
output$net.benefit$advantage=output$net.benefit$marker-output$net.benefit$all
#To view the table, simply call on the variable that it's stored in
Output
```

SAS Code

```
*Run the decision curve and save out net benefit results, specify xby=0.05 since we
want 5% increments;
%DCA(data=dca, outcome=cancer, predictors=marker, probability=no, xstart=0.05,
xstop=0.35, xby=0.05, graph=no, out= dcamarker);

*Load the data set with the net benefit results;
DATA dcamarker; SET dcamarker;
       *Calculate difference between marker and treat all;
       *Our standard approach is to biopsy everyone so this tells us how much better
       we do with the marker;
       advantage = marker - all
RUN;
```

The saved table lists the range of threshold probability that we specified, followed by the net benefits of treating all, none, our specified model, intervention avoided (we discuss this below), and the newly created variable which represent the increase in net benefits of our model using only the marker.

Net benefit has a ready clinical interpretation. The value of 0.03 at a threshold probability of 20% can be interpreted as: "Comparing to conducting no biopsies, biopsying on the basis of the marker is the equivalent of a strategy that found 3 cancers per hundred patients without conducting any unnecessary biopsies."

Interventions Avoided

As part of assessing the usefulness of the marker, we would be interested in whether using this marker to identify patients with and without cancer would help reduce unnecessary biopsies. This value was the "intervention avoided" column in the table that was saved out. To view it graphically, we would only need to specify it in our command.

Stata Code

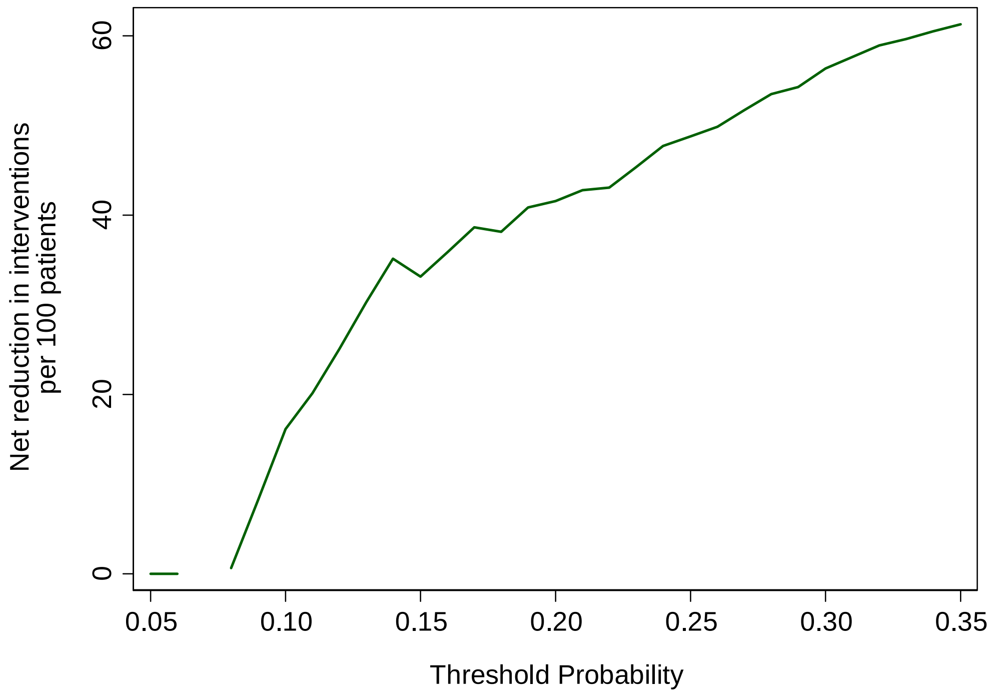
```
//Load the original data
use "dca.dta", clear
dca cancer marker, prob(no) intervention xstart(0.05) xstop(0.35)
```

R Code

```
dca(data=data.set, outcome="cancer", predictors="marker", probability=FALSE,
       intervention=TRUE, xstart=0.05, xstop=0.35)
```

SAS Code

```
%DCA(data=dca, outcome=cancer, predictors=marker, probability=no, intervention=yes,
xstart=0.05, xstop=0.35);
```



At a probability threshold of 15%, the net reduction in interventions is about 33 per 100 patients. In other words, at this probability threshold, biopsying patients on the basis of the marker is the equivalent of a strategy that reduced the biopsy rate by 33%, without missing any cancers.

Decision Curve Analysis for Survival Outcomes

Motivating Example

Continuing with the dataset of 750 patients who were biopsied, let us imagine a slightly different situation in which patients went to have the marker measured and patients were followed to determine whether they were eventually diagnosed with cancer, as well as the time to that diagnosis or censoring. We want to build a model of our own based on age, family history, and the marker, and assess how good the model is at predicting cancer by 18 months, 1.5 years.

The variables that will be used from the example dataset for this section of the tutorial are the following:

Variable	Description
cancer	Cancer Diagnosis: 0=No, 1=Yes
age	Patient Age, years
famhistory	Family History of Cancer: 0=No, 1=Yes
marker	Marker
dead	Died Prior to Possible Cancer Diagnosis
ttcancer	Years from Marker Measurement to Cancer Diagnosis/Censoring

Basic Data Set-up

We first need to declare our data to be survival-time data, with cancer considered the failure event.

Stata Code

```
//Declaring survival time data: follow-up time variable is ttcancer
//and the event is cancer
stset ttcancer, f(cancer)
```

R Code

```
#Source file to use stdca command
source("stdca.R")
#Creates a survival object with time to event variable as ttcancer and the event is
#cancer.
Srv = Surv(data.set$ttcancer, data.set$cancer)
```

SAS Code

```
*Creates a separate variable "t" for time so that "ttcancer" is not overwritten;
DATA stdca; SET home.origdca(RENAME=(ttcancer=_t));
       ttcancer = _t;
RUN;
```

The survival probability to any time-point can be derived from any type of survival model; here we use a Cox as this is the most common model in statistical practice. The formula for a survival probability from a Cox model is given by:

$$s(t|X) = s_0(t)\text{\^{}}\exp(X\beta)$$

Where X is matrix of covariates in the Cox model, β is a vector containing the parameter estimates from the Cox model, and $s_0(t)$ is the baseline survival probability to time t.

To get such values within our code, we will run a Cox model with age, family history, and the marker, as predictors, save out the baseline survival function in a new variable, and obtaining the linear prediction from the model for each subject.

We then obtain the baseline survival probability to our time point of interest. If no patient was observed at the exact time of interest, we can use the baseline survival probability to the observed time closest to, but not after, the time point. We can then calculate the probability of failure at the specified time point. For our example, we will use a time point of 1.5, which would corresponds to the eighteen months that we are interested in.

Stata Code

```
//Run the cox model and save out baseline survival in the “surv_func” variable
stcox age famhistory marker, basesurv(surv_func)
//get linear predictor for calculation of risk
predict xb, xb
//Obtain baseline survival at 1.5 years = 18 months
sum surv_func if _t <= 1.5
//We want the survival closest to 1.5 years
//This will be the lowest survival rate for all survival times ≤1.5
local base = r(min)
*Convert to a probability
g pr_failure18 = 1 - `base'^exp(xb)
label var pr_failure18 "Probability of Failure at 18 months"
```

R Code

```
#Load survival library
library(survival)
#Run the cox model
coxmod = coxph(Srv ~ age + famhistory + marker, data=data.set)
#the probability of failure is calculated by subtracting the probability of
#survival from 1.
data.set$pr_failure18 = c(1- (summary(survfit(coxmod,
      newdata=data.set), times=1.5)$surv))
```

SAS Code

```
*Run the Cox model;
PROC PHREG DATA=stdca;
       MODEL _t*cancer(0) = age famhistory marker;
       BASELINE OUT=baseline COVARIATES=stdca SURVIVAL=surv_func / NOMEAN METHOD=pl;
RUN;
*the probability of failure at 1.5 years is calculated by subtracting the
probability of survival from 1;
PROC SQL NOPRINT UNDO_POLICY=none;
       CREATE TABLE base_surv2 AS
       SELECT DISTINCT
              patientid, age, famhistory, marker, 1-min(surv_func) AS pr_failure18
       FROM baseline (WHERE=(_t<=1.5))
       GROUP BY patientid, age, famhistory, marker
       ;

       *merge survival estimates with original data;
       CREATE TABLE stdca AS
       SELECT A.*, B.pr_failure18
       FROM stdca A
              LEFT JOIN base_surv2 B
              ON (A.patientid=B.patientid) and (A.age=B.age) and
              (A.famhistory=B.famhistory) and (A.marker=B.marker);
       ;
QUIT;

DATA stdca; SET stdca;
       LABEL pr_failure18="Probability of Failure at 18 months";
RUN;
```

The code for running the decision curve analysis is straightforward after the probability of failure is calculated. All we have to do is specify the time point we are interested in. For our example, let us not only set the threshold from 0% to 50%, but also add smoothing. Note that different programs use different smoothers as there is no one smoother that is best in every situation. As such, results of a smoothed curve should always be compared with the unsmoothed curve to ensure accuracy.

Note to R users: A LOESS smoother is utilized in R. Infrequently, while performing the smoothing, R will print an error. In this case, you'll need to forego the automated smoothing within the DCA function. If you would like to perform smoothing, save the resulting net benefit values and apply your chosen method of smoothing after the decision curve has been calculated.

Note to SAS and R users: A LOESS smoother is utilized in SAS and R. The "smooth" option in Stata (used to produce the graphs shown) uses a different method for smoothing and therefore results may differ slightly.

Stata Code

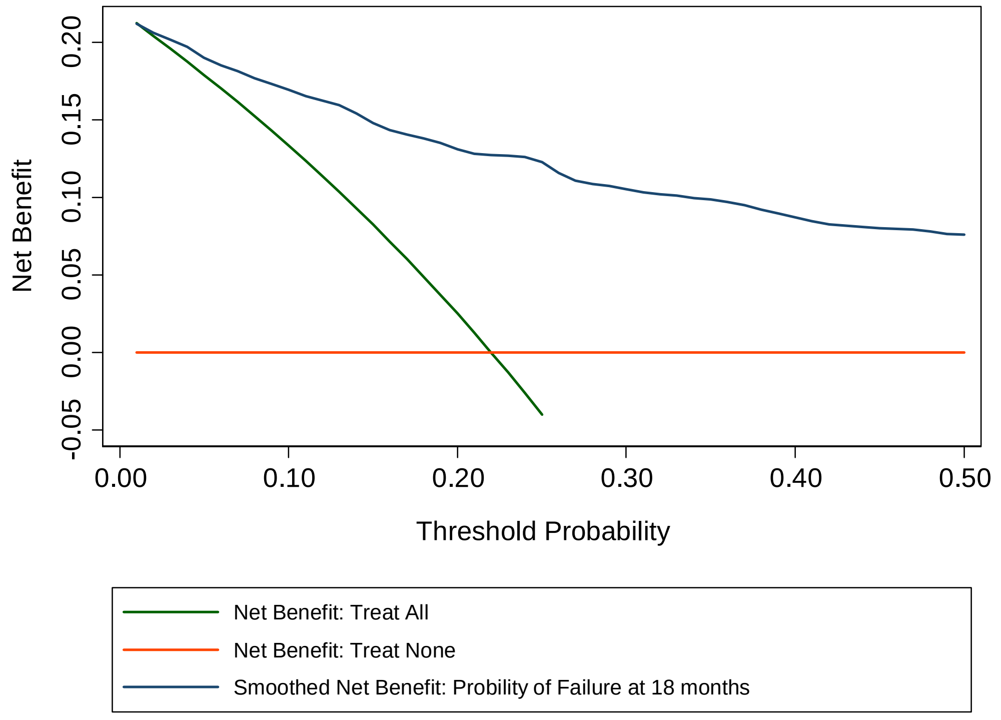
```
stdca pr_failure18, timepoint(1.5) xstop(0.5) smooth
```

R Code

```
stdca(data=data.set, outcome="cancer", ttoutcome="ttcancer", timepoint=1.5,
       predictors="pr_failure18", xstop=0.5, smooth=TRUE)
```

SAS Code

```
%STDCA(data=stdca, out=survivalmult, outcome=cancer, ttoutcome=ttcancer,
timepoint=1.5, predictors=pr_failure18, xstop=0.5);
```



This shows that using the model to inform clinical decisions will lead to superior outcomes for any decision associated with a threshold probability of above 2% or so.

Decision Curve Analysis with Competing Risks

At times, data sets are subject to competing risks. For example in our cancer dataset, patients may have died prior to cancer diagnosis. To run a competing risk analysis, we first create a failure variable that indicates which patients died before a cancer diagnosis. Using the traditional approach, patients are coded 0 if they do not have an event, 1 if they have the event of interest (cancer diagnosis) before the competing event and 2 if they have the competing event (death) before the event of interest.

Stata Code

```
g status = 0
replace status = 1 if cancer==1
replace status = 2 if cancer==0 & dead==1
```

R Code

```
data.set$status = data.set$cancer + data.set$dead * (data.set$cancer==0) * 2
```

SAS Code

```
DATA stdca; SET stdca;
       status = 0;
       IF cancer=1 THEN status=1;
       ELSE IF cancer=0 & dead=1 THEN status=2;
RUN;
```

We then again declare our data to be survival-time data and run the analysis specifying the competing risk variable.

Stata Code

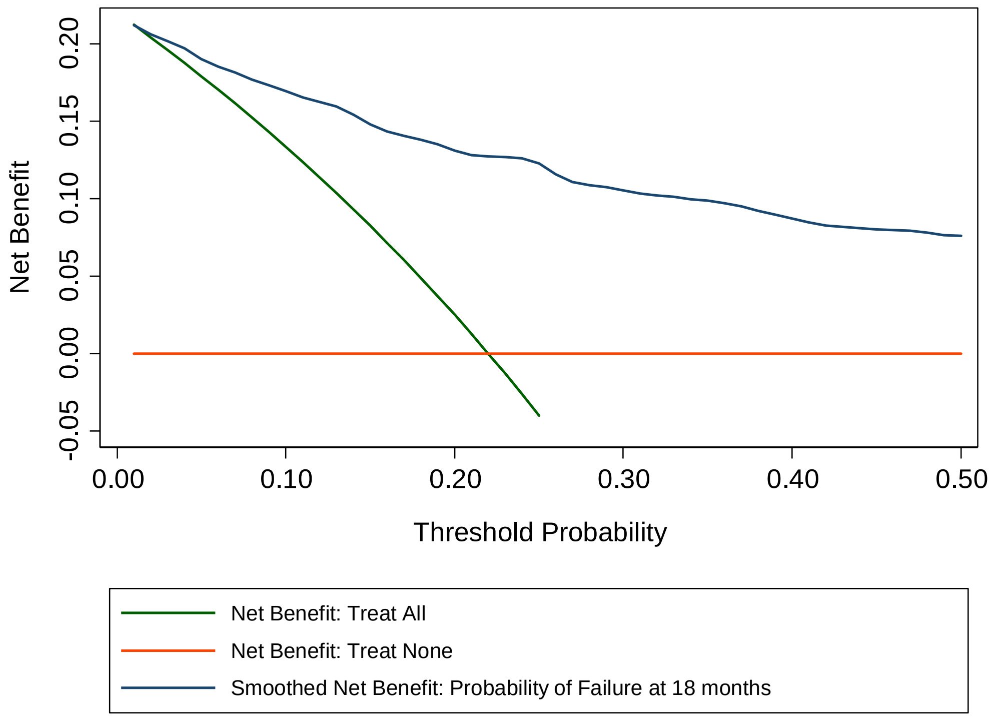
```
//We declare our survival data with the new event variable
stset ttcancer, f(status=1)
//Run the decision curve specifying the competing risk option
stdca pr_failure18, timepoint(1.5) compet1(2) smooth xstop(.5)
```

R Code

```
#We declare the survival data variables within the stdca function and run
#the decision curve analysis
stdca(data=data.set, outcome="status", ttoutcome="ttcancer", timepoint=1.5,
       predictors="pr_failure18", cmprsk=TRUE, smooth=TRUE, xstop=0.5)
```

SAS Code

```
*Run the decision curve specifying the competing risk option;
%STDCA(data=stdca, outcome=status, ttoutcome=ttcancer, timepoint=1.5,
predictors=pr_failure18, competerisk=yes, xstop=0.5);
```



The competing risk model does not change our conclusion that the statistical model would improve clinical decision making for all decisions.

As the two models look similar, we can show how to get both figures on one graph. This would also be a good example of how graphs can be drawn by saving out results. First, we start with recreating the original (Kaplan Meier) model again, but this time creating a temporary file that will be used to store the results. As we've already seen the figures after adding smoothing, let us now omit that.

Stata Code

```
stset ttcancer, f(cancer)
//Run the decision curve saving out the net benefits
stdca pr_failure18, timepoint(1.5) xstop(.5) nograph saving("km.dta", replace)
```

R Code

```
km = stdca(data=data.set, outcome="cancer", ttoutcome="ttcancer", timepoint=1.5,
       predictors="pr_failure18", xstop=0.5)
```

SAS Code

```
%STDCA(data=stdca, out=km, outcome=cancer, ttoutcome=ttcancer, timepoint=1.5,
predictors=pr_failure18, xstop=0.5);
```

Second, we will do the same for the second (Competing Risk) model and also saving these results to a different temporary file.

Stata Code

```
stset ttcancer, f(status=1)
stdca pr_failure18, timepoint(1.5) compet1(2) xstop(.5) nograph saving("cr.dta",
replace)
```

R Code

```
cr = stdca(data=data.set, outcome="status", ttoutcome="ttcancer", timepoint=1.5,
       predictors="pr_failure18", cmprsk=T, xstop=0.5)
```

SAS Code

```
%STDCA(data=stdca, out=cr, outcome=status, ttoutcome=ttcancer, timepoint=1.5,
predictors=pr_failure18, competerisk=yes, xstop=0.5);
```

Next, using the file with the standard decision curve, we will sort the values based on the threshold probabilities, rename and label the models so that we can tell that they are from the Kaplan-Meier model and save these changes.

Stata Code

```
use km.dta, clear
//Sort by the threshold so that we can merge later
sort threshold
//Rename the variables so that we know they are the Kaplan Meier estimates
rename pr_failure18 kmmodel
label var kmmodel "Kaplan-Meier: Pr(Failure) at 1.5 years"
rename all kmall
label var kmall "Kaplan-Meier: Treat All"
save kmsort.dta, replace
```

R Code

```
#No data manipulation required
```

SAS Code

```
*Sort by threshold variable;
PROC SORT DATA=km OUT=kmsort;
       BY threshold;
RUN;
*Rename the variables so that we know they are the Kaplan Meier estimates;
DATA kmsort; SET km(RENAME=(pr_failure18=kmmodel all=kmall));
       LABEL kmmodel="Kaplan-Meier: Pr(Failure) at 1.5 years";
       LABEL kmall="Kaplan-Meier: Treat All";
RUN;
```

Next we use the file with the results of the competing risk model and likewise sort by the probability threshold and rename variables, but instead of saving this as another file, let us simply merge the data with the data for the Kaplan-Meier decision curve models.

Stata Code

```
use cr.dta, clear
sort threshold
//Rename the variables so that we know they are the Competing Risk estimates
rename pr_failure18 crmodel
label var crmodel "Competing Risk: Pr(Failure) at 1.5 years"
rename all crall
label var crall "Competing Risk: Treat All"
merge 1:1 threshold using kmsort.dta
```

R Code

```
#No data manipulation required
```

SAS Code

```
*Sort by threshold variable;
PROC SORT DATA=cr OUT=crsort;
       BY threshold;
RUN;
*Rename the variables so that we know they are the Competing Risk estimates;
DATA crsort; SET crsort(RENAME=(pr_failure18=crmodel all=crall));
       LABEL crmodel="Competing Risk: Pr(Failure) at 1.5 years";
       LABEL crall="Competing Risk: Treat All";
RUN;
*Merge Kaplan-Meier and Competing Risk data using threshold probabilities;
DATA crsort;
       MERGE kmsort crsort;
       BY threshold;
RUN;
```

Now we simply create the graph. As the net benefits for the “Treat All” models extend far into negative values, let us specify as only being interest in the range of net benefits greater than −0.05. This is the default for the dca command, or it can be specified e.g “`ymin(-.01)`”. Let us also give the y-axis a title.

Stata Code

```
twoway (line kmall crall threshold if kmall>-0.05 & ///
       crall > -0.05, sort) || (line kmmodel crmodel none threshold, ///
       sort ytitle("Net Benefit"))
```

R Code

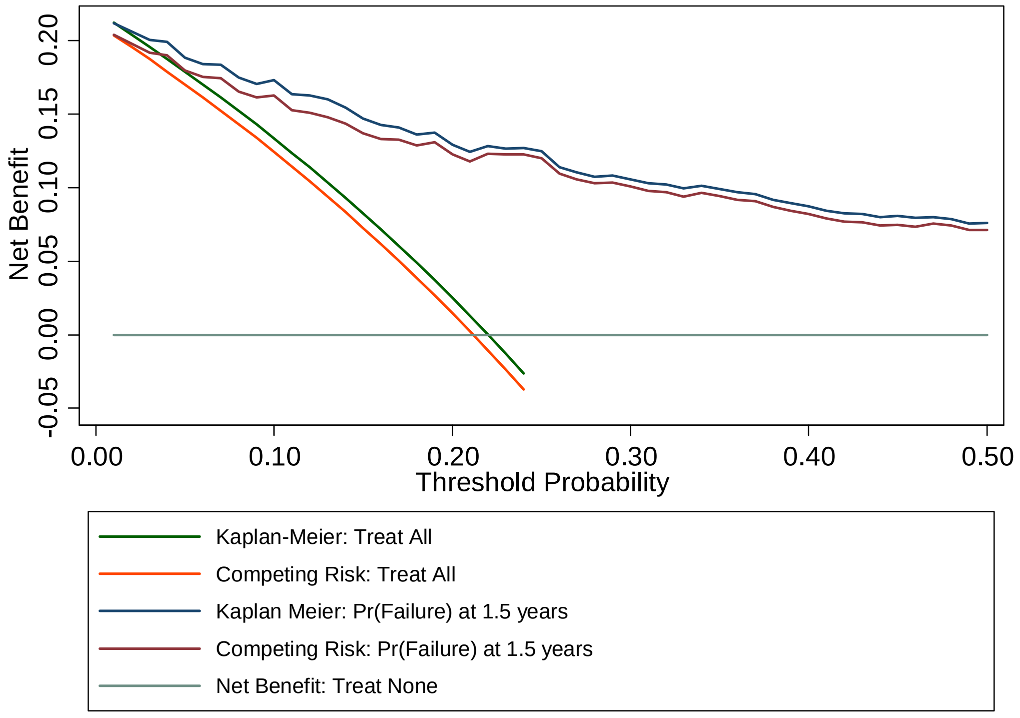
```
plot(km$net.benefit.threshold, km$net.benefit.none, type = "l", lwd=2,
       xlim=c(0,.50), ylim=c(-.05, .20), xlab = "Threshold Probability",
       ylab = "Net Benefit")
lines(km$net.benefit$threshold, km$net.benefit$all, type="l", col=8, lwd=2)
lines(km$net.benefit$threshold, cr$net.benefit$all, type="l", col=8, lwd=2, lty=2)
lines(km$net.benefit$threshold, km$net.benefit$pr_failure18, type="l", col=1)
lines(cr$net.benefit$threshold, cr$net.benefit$pr_failure18, type="l", col = 1,
       lty=2)
legend("topright", cex=0.8, legend=c("None", "KM All", "CR All", "KM Model", "CR
       Model"), col=c(17, 8, 8, 1, 1), lwd=c(2, 2, 2, 1, 1), lty=c(1, 1, 2, 1, 2))
```

SAS Code

```
PROC GPLOT DATA=crsort;
       axis1 ORDER=(-0.05 to 0.2 by 0.05) LABEL=(ANGLE=90 "Net Benefit") MINOR=none;
       axis2 ORDER=(0.0 to 0.5 by 0.1) LABEL=("Threshold Probability") MINOR=none;

       legend1 LABEL=none ACROSS=1 DOWN=5 POSITION=(bottom center outside)
       CBORDER=black VALUE=("KM: Treat All" "CR: Treat All" "Pr(Failure) at 1.5 Years
       (KM)" "Pr(Failure) at 1.5 Years (CR)" "None");

       PLOT    kmall*threshold
               crall*threshold
               kmmodel*threshold
               crmodel*threshold
               none*threshold / OVERLAY VAXIS=axis1 HAXIS=axis2 LEGEND=legend1;
               SYMBOL INTERPOL=JOIN;
RUN;
QUIT;
```



Here the decision curves adjusting for competing risk are shown in orange and red; the decision curves with patients censored at the time of death are shown in green and blue. Competing risk shifts all lines down, that is, estimates a lower net benefit, because censoring at the time of death overestimates the risk of death.

Assessing Clinical Utility in a Case-Control Design

The problem with applying decision curve analysis to case-control data is that net benefit depends on prevalence, and prevalence in case-control studies is fixed by design. This problem can be solved by use of recalibration (sometimes called a "Bayes factor").

Let us assume that for our example, there was a variable ("casecontrol") which denotes whether the patient was a part of the case control study to assess whether family history along with age can predict cancer. The matching was 3:1, such that the prevalence of cancer in the study was 25%. We will assume that the true risk of cancer was 5%. We will first build a model using the data set of only the patients in the case control study and then obtain the log odds of disease (linear predictor or fitted values).

Stata Code

```
//Load our original data
use "dca.dta", clear
//Use only the data from the case control study
drop if casecontrol == 0
//Create the model
logit cancer famhistory age
//Save out the linear predictor, rather than the probability
predict xb, xb
```

R Code

```
#Use only the data from the case control study
casecontrol = subset(data.set, casecontrol ==1)
#Create the model
model = glm(cancer~ age + famhistory, family=binomial(link="logit"),
       data=casecontrol)
#Save out the linear predictor
xb = predict(model)
```

SAS Code

```
*Load our original data and use only the data from the case control study;
DATA cc_dca; SET home.origdca;
       IF casecontrol=0 THEN DELETE;
       ELSE OUTPUT;
RUN;
*Create the model and save out the linear predictor;
PROC LOGISTIC DATA=cc_dca;
       MODEL cancer = famhistory age;
       OUTPUT OUT=cc_dca XBETA=xb;
RUN;
```

We can then add the Bayes factor to the linear predictor, which is the log of the true odds of cancer divided by the odds of cancer in the case-control study.

Stata Code

```
//The true risk stored in a local
local true = 0.05
sum cancer
//The observed risk, which is the mean of our data, is stored in a local
local design = r(mean)
//The Bayes factor is stored in a local
local Bayes=log((`true'/(1-`true'))/(`design'/(1-`design')))
//We add the Bayes factor to the linear predictor
replace xb=xb+`Bayes'
```

R Code

```
#The true risk is stored in a scalar
true = 0.05
#The observed risk, the mean of our data, is stored in a scalar
design = mean(casecontrol$cancer)
#The Bayes factor is stored in a scalar
Bayes = log((true/(1-true))/(design/(1-design)))
#We add the Bayes factor to the linear predictor
xb = xb+Bayes
```

SAS Code

```
*the true risk is stored as a macro variable;
%LET true = 0.05;
*The observed risk, which is the mean of our data, is stored in a macro variable;
PROC MEANS DATA=cc_dca;
       VAR cancer;
       OUTPUT OUT=risk MEAN=risk;
RUN;
DATA _NULL_; SET risk;
       CALL SYMPUT("design",risk);
RUN;
*The Bayes factor is stored as a macro variable;
%LET Bayes = (%SYSEVALF(&true.)/(1-%SYSEVALF(&true.)))/(%SYSEVALF(&design.)/(1-
%SYSEVALF(&design.)));
%LET Bayes = %SYSEVALF(%SYSFUNC(log(&Bayes.)));
*We add the Bayes factor to the linear predictor;
DATA cc_dca; SET cc_dca;
       adj_xb = xb + %SYSEVALF(&Bayes.);
RUN;
```

This is then converted to a probability. We use this probability both as the predictor and the outcome for the decision curve, using the assumption that the model is well-calibrated (note that no other assumption is possible: we can’t assess calibration in a case control study).

Stata Code

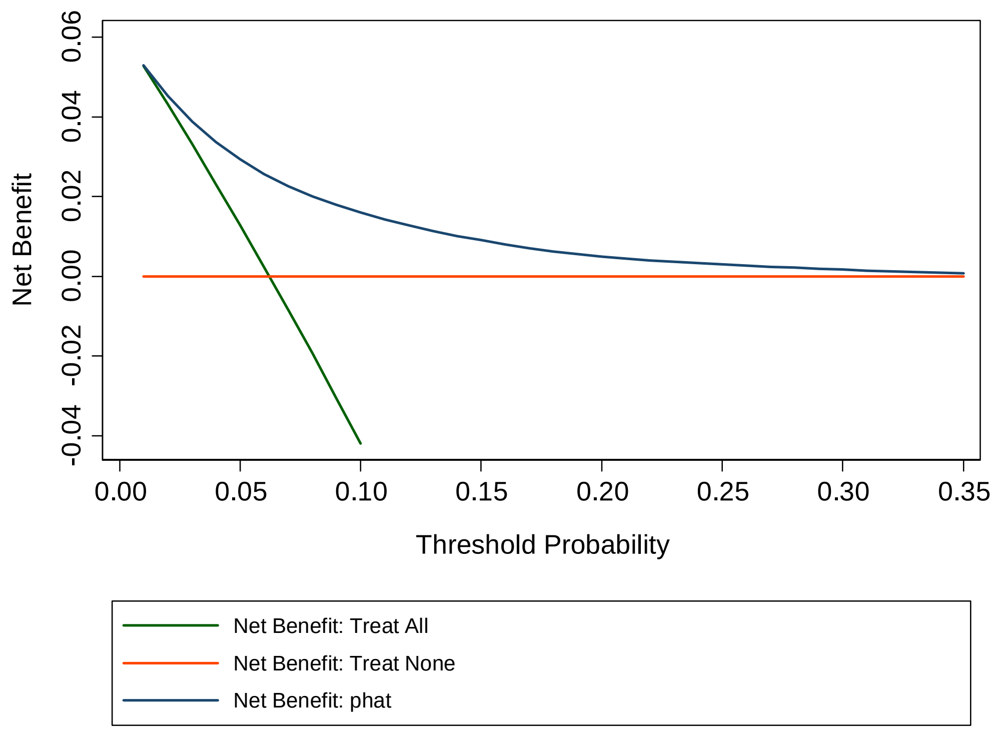
```
//Convert to a probability
g phat=invlogit(xb)
//Run the decision curve
dca phat phat, xstop(.35) xlabel(0(0.05)0.35)
```

R Code

```
#Convert to a probability
casecontrol$phat = exp(xb)/(1+exp(xb))
#Run the decision curve
dca(data=casecontrol, outcome="phat", predictors="phat", xstop=0.35)
```

SAS Code

```
*Convert to a probability;
DATA cc_dca; SET cc_dca;
       phat = exp(adj_xb)/(1+exp(adj_xb));
RUN;
*Run the decision curve;
%DCA(data=cc_dca, out=test, outcome=phat, predictors=phat, xstop=0.35,
probability=yes);
```



This decision curve can be used to evaluate whether a statistical model based on family history of cancer and age should be used to determine the intensity of cancer screening. The decision curve shows that the statistical model would be helpful for decisions with a threshold probability of 2% or above. In other words, in the unlikely event that a patient would agree to be screened even if he or she had a 1% lifetime risk of cancer, they should be screened irrespective of family history of cancer and age; otherwise, screening should be based on the risks from the model.

Correction for Overfit

As the model created from a dataset may include random error (noise) instead of just the relationships between the outcome and predictors, we should take care to correct for overfit when comparing it to other models. One such way is by using repeated 10-fold cross validation. The main steps are as follows:

1. Randomly divide the dataset into 10 groups of equal size, with equal numbers of events in each group.
2. Fit the model using all but the first group
3. Apply the model created from all but the first group (in step 2), to the first group to obtain the predicted probability of the event.
4. Repeat steps (2) and (3) leaving out and then applying the fitted model for each of the groups. Every subject now has a predicted probability of the event.
5. Using the predicted probabilities, compute the net benefit at various threshold probabilities.

One approach is to repeat this process many times and take an average. Although this step is not always done, we will include code for it here:

6. Repeat steps (1) to (5) 200 times. The corrected net benefit for each threshold probability is the mean across the 200 replications.

Let us begin by writing an overall loop which will be used to run the optional, repeated 10-fold cross validation, which for our example will be 200 iterations. For simplicity, to exclude the multiple cross validation in this code we only need to declare the end of the sequence of the forloop as 1 instead of 200. Alternatively, we could exclude the forloop command and simply take care to rename any reference to the loop.

Continuing with our example, we will also need to create two variables which will later be used to store the probabilities from our prediction models. For our example, in particular, we are interested in comparing a model incorporating age, family history, and marker (let us call this the “full” model) and we wish to compare that model to one only including age and family history (which we’ll refer to as the “base” model). Code can easily be extended for more than two models, or minimized for only one model, but for our example, two will do.

Stata Code
```
//Load Original Dataset
use "dca.dta", clear
//To skip this optional loop used for running the cross validation multiple times
//either 1) change it to “forvalues i=1(1)1 {” or
//2) omit this line of code and take care to change any code which references “i”
forvalues i=1(1)200 {
    //Local macros to store the names of model.
    local prediction1 = "base"
    local prediction2 = "full"
    //Create variables to later store probabilities from each prediction model
    quietly g `prediction1'=.
    quietly g `prediction2'=.
```

R Code
```
#To skip this optional loop used for running the cross validation multiple times
#either 1) change it to “for (i in 1:1) {” or
#2) omit this line of code and take care to change any code which references “i”
for (i in 1:200) {
    #Create variables to later store probabilities from each prediction model
    data.set$pred1=NA
    data.set$pred2=NA
```
SAS Code
```
*In SAS, we will use a macro to do the ten-fold cross-validation. This code is
provided at the end of this example.;
```

Next, we will begin with step 1, which for our example entails randomly assigning our patients to one of 10 groups and ensuring that equal numbers of patients with cancer are in each group.

Stata Code
```
//Create a variable to be used to ‘randomize’ the patients.
quietly g u = uniform()
//Sort by the event to ensure equal number of patients with the event are in each
//group
sort cancer u
//Assign each patient into one of ten groups
g group = mod(_n, 10) + 1
```
R Code
```
#Create a variable to be used to ‘randomize’ the patients.
u = runif(750,0,1)
#Sort by event to ensure equal number of patients with the event are in each group
data.set = data.set[order(cancer, u),]
#Assign each patient into one of ten groups
data.set$group = (seq(1:dim(data.set)[1]) %% 10)+1
```

We then move on to writing the loop for steps 2-4, by building the model using all the data excluding one group, applying the created model to the group that was excluded in order so as to predict the probability of the event (cancer) and storing that value. Once this is done for each group, there will be predicted probabilities for each patient based on a model that did not utilize their data when creating the mode. We will do this for each model of interest: the base and full model.

Stata Code

```
//Loop through to run through for each of the ten groups
forvalues j=1(1)10 {
       //First for the "base" model:
       //Fit the model excluding the jth group.
       quietly logit cancer age famhistory if group!=`j'
       //Predict the probability of the jth group.
       quietly predict ptemp if group==`j'
       //Store the predicted probabilities of the jth group (that was not used in
       //creating the model) into the variable previously created
       quietly replace `prediction1`num'' = ptemp if group==`j'
       //Dropping the temporary variable that held predicted probabilities for all
       //patients
       drop ptemp
       //Likewise, for the second "final" model
       quietly logit cancer age famhistory marker if group!=`j'
       quietly predict ptemp if group==`j'
       quietly replace `prediction2' = ptemp if group==`j'
       drop ptemp
}
```

R Code

```
#As R predicts probabilities based on the data that was used to create the model,
#we will need to calculate the probabilities ourselves.
#Set the constant term as 1.
#Create variables to store the log of the odds Xβ, and hold the model coefficients
data.set$xb1=NA
data.set$xb2=NA
data.set$cons=1
mod1=NULL
mod2=NULL
#Loop through to run through for each of the ten groups
for (j in 1:10) {
       #First for the “base” model:
       #Predict the probability of the jth group.
       mod1[[j]] = glm(cancer~age+famhistory, data=subset(data.set,group!=j),
       family=binomial)
       #Calculating and storing Xß
       data.set$xb1[data.set$group==j]=data.matrix(data.set[data.set$group==j,
       c("cons","age","famhistory")]) %*% mod1[[j]]$coef
       #Likewise, for the second “final” model
       mod2[[j]] = glm(cancer~age+famhistory+marker, data=subset(data.set,group!=j),
       family=binomial)
       data.set$xb2[data.set$group==j]=data.matrix(data.set[data.set$group==j,
       c("cons","age","famhistory", "marker")]) %*% mod2[[j]]$coef
}
#Now calculate the probability of having the event for each model
data.set$pred1 = 1/(1+exp(-1*(data.set$xb1)))
data.set$pred2 = 1/(1+exp(-1*(data.set$xb2)))
```

Next we will invoke the dca command to compute the net benefits (step 5) and save our results to a temporary file. Note that if we had excluded the optional multiple cross validation step, the results of invoking this command would be the net benefits (and interventions) corrected for overfitting. For our example, as we are including the multiple 10 fold cross validation step, we will suppress the current graph as our interest lies in the mean of the corrected net benefits, which requires additional lines of code. Having (temporarily) saved the data, we will drop the created variables and close the overall loop so that this process can be iterated a total of 200 times. After the iterations of the multiple cross validation, we will combine the results of all 200 iterations and take the mean of all the result, thus obtaining the corrected net benefits.

Stata Code

```
//Creating a temporary file to store the results of each of the iterations of our
//decision curve for the multiple the 10 fold cross validation
//This step may omitted if the optional forvalues loop was excluded.
tempfile dca`i'
//Run decision curve, and save the results to the tempfile.
//For those excluding the optional multiple cross validation, this decision curve
//(to be seen by excluding "nograph") and the results (saved under the name of your
//choosing) would be the decision curve corrected for overfit.
quietly dca cancer `prediction1' `prediction2', xstop(.5) nograph ///
       saving("`dca`i''")
drop u group `prediction1' `prediction2'
} //This closing bracket ends the initial loop for the multiple cross validation.
//It is also necessary for those who avoided the multiple cross validation
//by changing the value of the forvalues loop from 200 to 1*/
//The following is only used for the multiple 10 fold cross validations.
       use "`dca1'", clear
       forvalues i=2(1)200 {
              //Append all values of the multiple cross validations into the first
              //file
              append using "`dca`i''"
       }
       //Calculate the average net benefit across all iterations of the multiple
       //cross validation
       collapse all none base full base_i full_i, by(threshold)
       save "Cross Validation DCA Output.dta", replace
```

R Code

```
#Running the decision curve.
#For those excluding the optional multiple cross validation, this decision curve
#to be seen by excluding “graph=FALSE”) and the results (saved under the name of
your #choosing) would be the decision curve corrected for overfit.
output = dca(data=data.set, outcome="cancer", predictors=c("pred1", "pred2"),
       graph=FALSE, xstop=.5)
#The following is only used for the multiple 10 fold cross validations.
       if(i==1){
              dcaoutput = output$net.benefit
              } else{
              #Append all result of the multiple cross validations into the first
              #file
              dcaoutput = rbind(dcaoutput, output$net.benefit)
              }
} #This closing bracket ends the initial loop for the multiple cross validation.
#It is also necessary for those who avoided the multiple cross validation by
#changing the value of the for loop from 200 to 1
       #Only used for the multiple cross validation. Calculate the average net
       #benefit across all iterations.
       data=aggregate(dcaoutput, by=list(dcaoutput$threshold), mean)[-1]
```

With the mean corrected net benefits calculated, we simply need to plot and label the curves. Again, as the net benefits for the “Treat All” models extend far into negative values, let us specify as only being interest in the range of net benefits greater than −0.05.

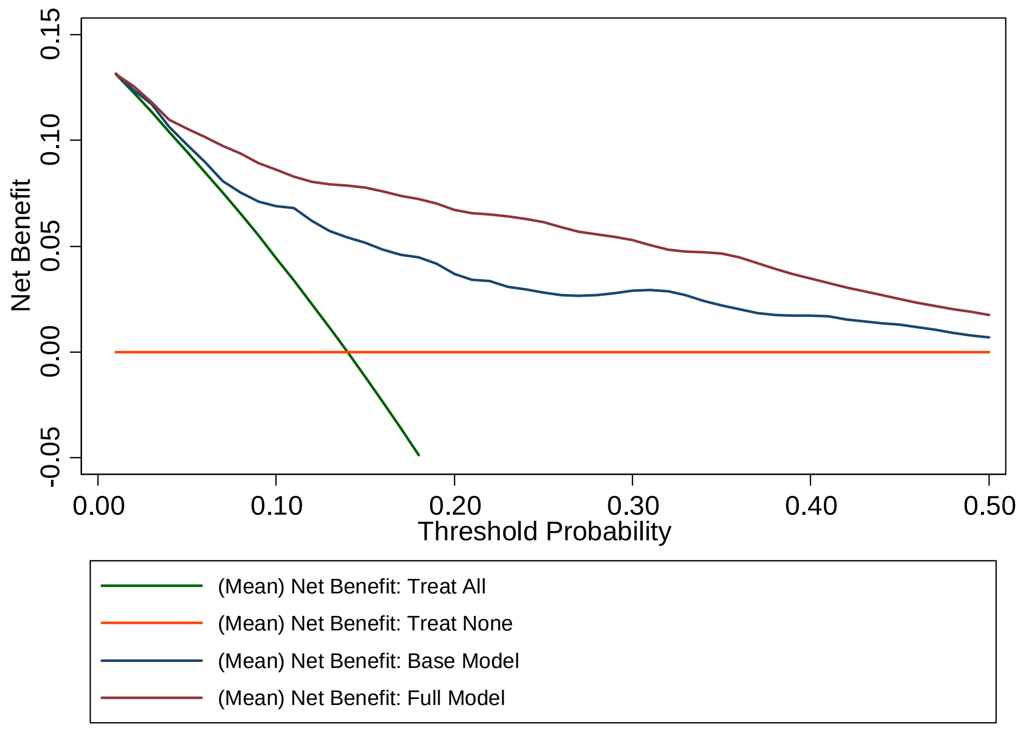
Note that since the groups used for the 10-fold cross validation are created by a random number generator, the resulting net benefit values and graph may not be exactly the same if we were to run this method again, yet the results would be similar.

Stata Code

```
//Labeling the variables so that the legend will have the proper labels
label var all "(Mean) Net Benefit: Treat All"
label var none "(Mean) Net Benefit: Treat None"
label var base "(Mean) Net Benefit: Base Model"
label var full "(Mean) Net Benefit: Full Model"
label var base_i "(Mean) Intervention: Base Model"
label var full_i "{Mean) Intervention: Full Model"
//Plotting the figure of all the net benefits.
twoway (line all threshold if all>-0.05, sort) || (line none base full threshold,
sort)
```

R Code

```
#Plotting the aces and “Treat None” Model
plot(data$threshold, data$none, type="l", xlim=c(0, 0.50), ylim=c(-0.05, 0.15),
      lwd=2, xlab="Threshold Probability", ylab="Net Benefit")
#Plotting the “Treat All” Model
lines(data$threshold, data$all, type="l", col=8, lwd=2)
#Plotting the “Base” (pred1: only using age and family history) Model
lines(data$threshold, data$pred1, type="l", col=2, lwd=2)
#Plotting the “Full” (pred2: using the marker, age and family history) Model
lines(data$threshold, data$pred2, type="l", col=1, lwd=2)
# Adding a legend to distinguish each of the models.
legend("topright", cex=0.8, legend=c("(Mean) Net Benefit: Treat None", "(Mean) Net
      Benefit: Treat All", "(Mean) Net Benefit: Base Model", "(Mean) Net Benefit:
      Full Model"), col=c(17, 8, 2, 1), lwd=c(2, 2, 2, 2))
```



This graph, displaying all the models after correcting for overfit, shows that the full model—which includes age, family history and the marker— leads to the highest net benefit, with the exception for the threshold probability below 3%.

SAS Code – Ten-Fold Cross Validation

```
%MACRO CROSSVAL;

*To skip the optional loop used for running the cross validation multiple times,
either 1) change it to "%DO x = 1 %TO 1" or 2) omit this line of code and take care to
change any code which references "&x.";

%DO x = 1 %TO 200;

       *Load original data and create a variable to be used to 'randomize' patients;
       DATA dca_of; SET home.origdca;
              u = RAND("Uniform");
       RUN;

       *Sort by the event to ensure equal number of patients with the event are in
       each group;
       PROC SORT DATA=dca_of;
              BY cancer u;
       RUN;

       *Assign each patient into one of ten groups;
       DATA dca_of; SET dca_of;
              group=MOD(_n_,10) + 1;
       RUN;

       *Loop through to run through for each of the ten groups;
       %DO y = 1 %TO 10;

              *First for the "base" model, fit the model excluding the yth group.;
              PROC LOGISTIC DATA=dca_of OUTMODEL=base&y. DESCENDING NOPRINT;
                     MODEL cancer = age famhistory;
                     WHERE group ne &y.;
              RUN;

              *Put yth group into base test dataset;
              DATA basetest&y.; SET dca_of;
                     WHERE group = &y.;
              RUN;

              *Apply the base model to the yth group and save the predicted
              probabilities of the yth group (that was not used in creating the model);
              PROC LOGISTIC INMODEL=base&y. NOPRINT;
                     SCORE DATA=basetest&y. OUT=base_pr&y.;
              RUN;

              *Likewise, for the second "final" model, fit the model excluding the yth
              group;
              PROC LOGISTIC DATA=home.dca_of OUTMODEL=final&y. DESCENDING NOPRINT;
                     MODEL cancer = age famhistory marker;
                     WHERE group ne &y.;
              RUN;

              *Put yth group into final test dataset;
              DATA finaltest&y.; SET dca_of;
                     WHERE group = &y.;
              RUN;

              *Apply the final model to the yth group and save the predicted
              probabilities of the yth group (that was not used in creating the model);
              PROC LOGISTIC INMODEL=final&y. NOPRINT;
                     SCORE DATA=finaltest&y. OUT=final_pr&y.;
              RUN;
```

```
        %END;

        *Combine base model predictions for all 10 groups;
        DATA base_pr(RENAME=(P_1=base_pred));
               SET base_pr1-base_pr10;
        RUN;

        *Combine final model predictions for all 10 groups;
        DATA final_pr(RENAME=(P_1=final_pred));
               SET final_pr1-final_pr10;
        RUN;

        *Sort and merge base model and final model prediction data together;
        PROC SORT DATA=base_pr NODUPKEYS;
               BY patientid;
        RUN;

        PROC SORT DATA=final_pr NODUPKEYS;
               BY patientid;
        RUN;

        DATA all_pr;
               MERGE base_pr final_pr;
               BY patientid;
        RUN;

        *Run decision curve and save out results;
        *For those excluding the optional multiple cross validation, this decision
        curve (to be seen by using "graph=yes") and the results (saved under the name
        of your choosing) would be the decision curve corrected for overfit;

        %DCA(data=all_pr, out=dca&x., outcome=cancer, predictors=base_pred final_pred,
        graph=no, xstop=0.5);

*This "%END" statement ends the initial loop for the multiple cross validation. It is
also necessary for those who avoided the multiple cross validation by changing the
value in the DO loop from 200 to 1;
%END;

*The following is only used for the multiple 10 fold cross validation.;

*Append all values of the multiple cross validation;
DATA _NULL_;
        CALL SYMPUTX("n",&x.-1);
RUN;

DATA allcv_pr; SET dca1-dca&n.;
RUN;

*Calculate the average net benefit across all iterations of the multiple cross
validation;
PROC MEANS DATA=allcv_pr NOPRINT;
        CLASS threshold;
        VAR all base_pred final_pred;
        OUTPUT OUT=minfinal MEAN=all base_pred final_pred;
RUN;
```

```
*Save out average net benefit and label variables so that the plot legend will have
the proper labels.;
DATA allcv_pr(KEEP=base_pred final_pred all none threshold);
       SET allcv_pr(DROP=base_pred final_pred all) minfinal;
       LABEL all="(Mean) Net Benefit: Treat All";
       LABEL none="(Mean) Net Benefit: Treat None";
       LABEL base_pred="(Mean) Net Benefit: Base Model";
       LABEL final_pred="(Mean) Net Benefit: Full Model";
       RUN;

%MEND CROSSVAL;

*Run the crossvalidation macro;
%CROSSVAL;

*Plotting the figure of all the net benefits;
PROC GPLOT DATA=allcv_pr;
       axis1 ORDER=(-0.05 to 0.15 by 0.05) LABEL=(ANGLE=90 "Net Benefit") MINOR=none;
       axis2 ORDER=(0 to 0.5 by 0.1) LABEL=("Threshold Probability") MINOR=none;
       legend LABEL=NONE ACROSS=1 CBORDER=BLACK;

       PLOT   all*threshold
              none*threshold
              base_pred*threshold
              final_pred*threshold /
              VAXIS=axis1
              HAXIS=axis2
              LEGEND=legend OVERLAY;
              SYMBOL INTERPOL=JOIN;
RUN;
QUIT;
```