Causal modelling vs Predictive Modelling Tutorial

Dr Maposa

The Data

The data constitute 2500 participants with responses on 8 variables . *cancer* is the outcome of interest. The goal of study was to assess the effect of HbA1c on cancer outcomes. The other characteristics, socioeconomic position (***SEP),*** *ethnicity* ***(ethnicity)*** *index, white or other (****BME****) is binaryand deprivation (****deprivation****) is rank from least deprived 1 to most deprived etc.* We are going to use both causal statistical modelling and predictive modelling show casing their strengths and weaknesses as discussed in lecture. Next we load the data and load the relevant R packages.

Packages and dataset

```
library(knitr)    # For knitting document and include_graphics function
library(ggplot2)  # For plotting
library(png)
library(gmodels)
library(randomForest)
```

```
randomForest 4.7-1.1
```

```
Type rfNews() to see new features/changes/bug fixes.
```

```
Attaching package: 'randomForest'
```

```
The following object is masked from 'package:ggplot2':

    margin
```

```
library(caret)
```

```
Loading required package: lattice
```

```
library(mlbench)
require(xgboost)
```

```
Loading required package: xgboost
```

```
library(pROC)
```

```
Type 'citation("pROC")' for a citation.


Attaching package: 'pROC'

The following object is masked from 'package:gmodels':

    ci

The following objects are masked from 'package:stats':

    cov, smooth, var
```

```
library(tidymodels)
```

```
-- Attaching packages ------------------------------------- tidymodels 1.2.0 --

v broom        1.0.6     v rsample      1.2.1
v dials        1.3.0     v tibble       3.2.1
v dplyr        1.1.4     v tidyr        1.3.1
v infer        1.0.7     v tune         1.2.1
v modeldata    1.4.0     v workflows    1.1.4
v parsnip      1.2.1     v workflowsets 1.1.0
v purrr        1.0.2     v yardstick    1.3.1
v recipes      1.1.0
```

```
-- Conflicts --------------------------------------- tidymodels_conflicts() --
x dplyr::combine()         masks randomForest::combine()
x purrr::discard()         masks scales::discard()
x dplyr::filter()          masks stats::filter()
x dplyr::lag()             masks stats::lag()
x purrr::lift()            masks caret::lift()
x randomForest::margin()   masks ggplot2::margin()
x yardstick::precision()   masks caret::precision()
x yardstick::recall()      masks caret::recall()
x yardstick::sensitivity() masks caret::sensitivity()
x dplyr::slice()           masks xgboost::slice()
x yardstick::specificity() masks caret::specificity()
x recipes::step()          masks stats::step()
* Search for functions across packages at https://www.tidymodels.org/find/
```

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v forcats   1.0.0     v readr     2.1.5
v lubridate 1.9.3     v stringr   1.5.1

-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x readr::col_factor()    masks scales::col_factor()
x dplyr::combine()       masks randomForest::combine()
x purrr::discard()       masks scales::discard()
x dplyr::filter()        masks stats::filter()
x stringr::fixed()       masks recipes::fixed()
x dplyr::lag()           masks stats::lag()
x purrr::lift()          masks caret::lift()
x randomForest::margin() masks ggplot2::margin()
x dplyr::slice()         masks xgboost::slice()
x readr::spec()          masks yardstick::spec()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library("dplyr")                                # Load dplyr
#library("rms")
library("DALEX")
```

```
Welcome to DALEX (version: 2.4.3).
Find examples and detailed introduction at: http://ema.drwhy.ai/
```

```
Attaching package: 'DALEX'

The following object is masked from 'package:dplyr':

    explain
```

```
library(nlpred)
```

```
Loading required package: data.table

Attaching package: 'data.table'

The following objects are masked from 'package:lubridate':

    hour, isoweek, mday, minute, month, quarter, second, wday, week,
    yday, year

The following object is masked from 'package:purrr':

    transpose

The following objects are masked from 'package:dplyr':

    between, first, last
```

```
suppressWarnings(suppressMessages(library(readxl)))
suppressWarnings(suppressMessages(library(curl)))
suppressWarnings(suppressMessages(library(vtable)))
suppressWarnings(suppressMessages(library(caTools)))
suppressWarnings(suppressMessages(library(caret)))
suppressWarnings(suppressMessages(library(qwraps2)))
suppressPackageStartupMessages(library("gtsummary"))
suppressPackageStartupMessages(library("dplyr"))
suppressPackageStartupMessages(library("kableExtra"))
suppressPackageStartupMessages(library("rcompanion"))
suppressPackageStartupMessages(library("ggplot2"))
suppressPackageStartupMessages(library("ggsignif"))
suppressPackageStartupMessages(library("ggdag"))
popdatex <- read_excel("popdatex.xlsx")
head(popdatex, 5)
```

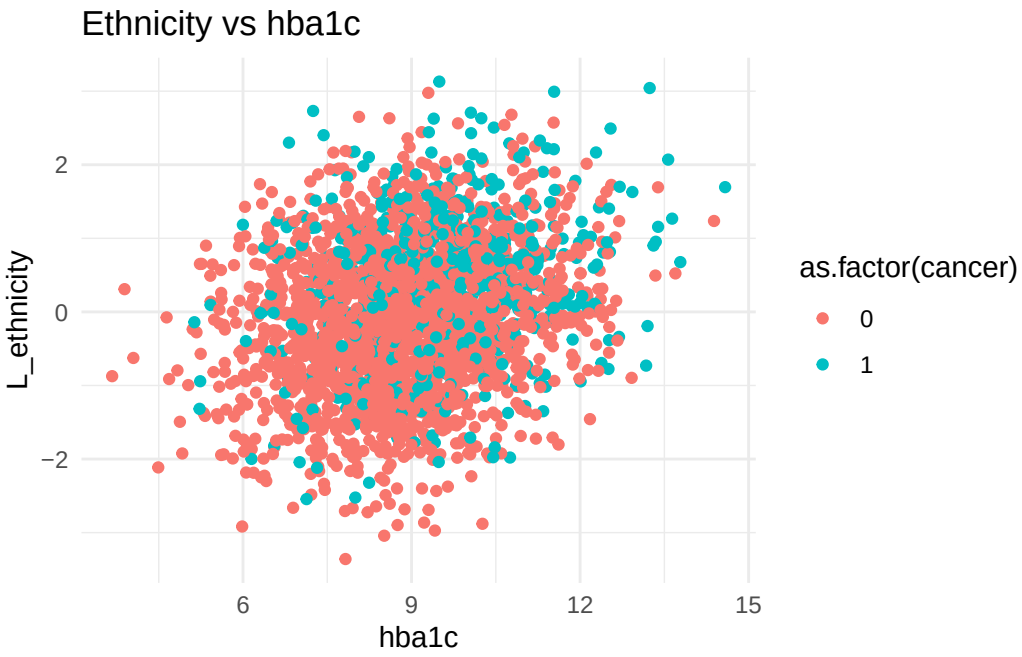
```
# A tibble: 5 x 8
    L_SEP L_ethnicity cancer hba1c sample    id   BME deprived
    <dbl>       <dbl>  <dbl> <dbl>  <dbl> <dbl> <dbl>    <dbl>
1  1.11       -0.218       0  9.31      1     1     0        5
2 -0.206       2.29        1 10.7       1     2     1        3
3  1.22       -0.0640      1 11.1       1     3     0        5
4  0.0993     -0.692       1  9.68      1     4     0        3
5  1.51       -1.38        0  9.30      1     5     0        5
```

```
sumtable(round(popdatex, 1), out="return", digits=2)
```

```
     Variable    N   Mean Std. Dev.  Min Pctl. 25 Pctl. 75  Max
1       L_SEP 2500 -0.012         1 -3.7     -0.7      0.7  3.5
2 L_ethnicity 2500 -0.024         1 -3.4     -0.7      0.7  3.1
3      cancer 2500   0.25      0.43    0        0        0    1
4       hba1c 2500      9       1.5  3.7      7.9      9.9   15
5      sample 2500   0.75      0.43    0        1        1    1
6          id 2500   1250       722    1      626     1875 2500
7         BME 2500   0.25      0.43    0        0        1    1
8    deprived 2500      3       1.4    1        2        4    5
```

Explore the data to see distribution of cancer status given covariates

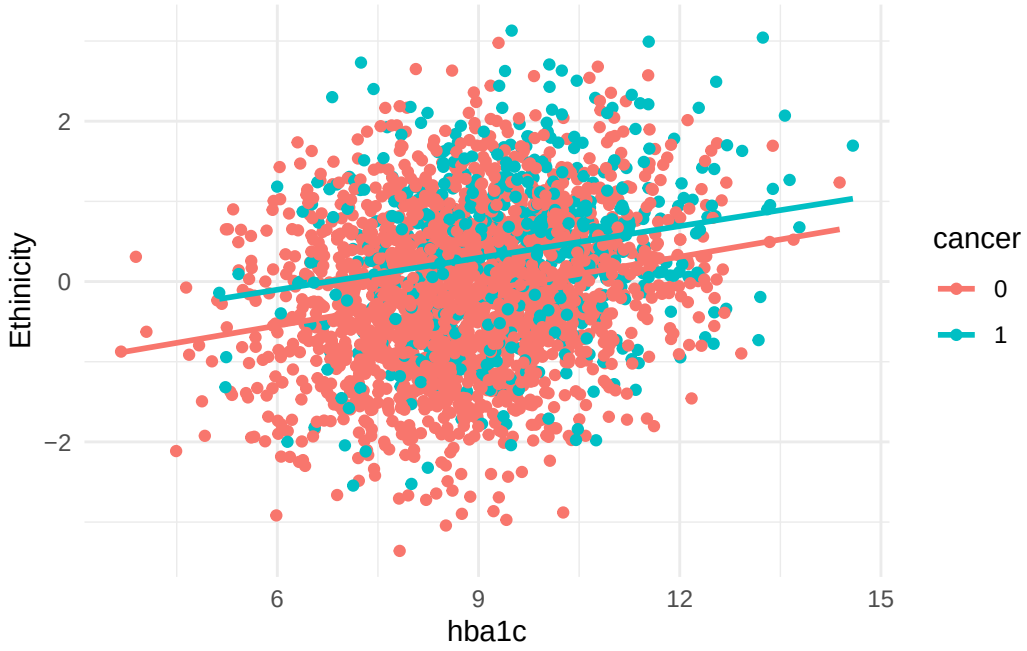
```
# Create a basic scatter plot
ggplot(
  data = popdatex,
  aes(x = hba1c, y = L_ethnicity, color = as.factor(cancer))) +
  geom_point() +
  labs(title = "Ethnicity vs hba1c",
       x = "hba1c",
       y = "L_ethnicity") +
  theme_minimal()
```



- Another visualization

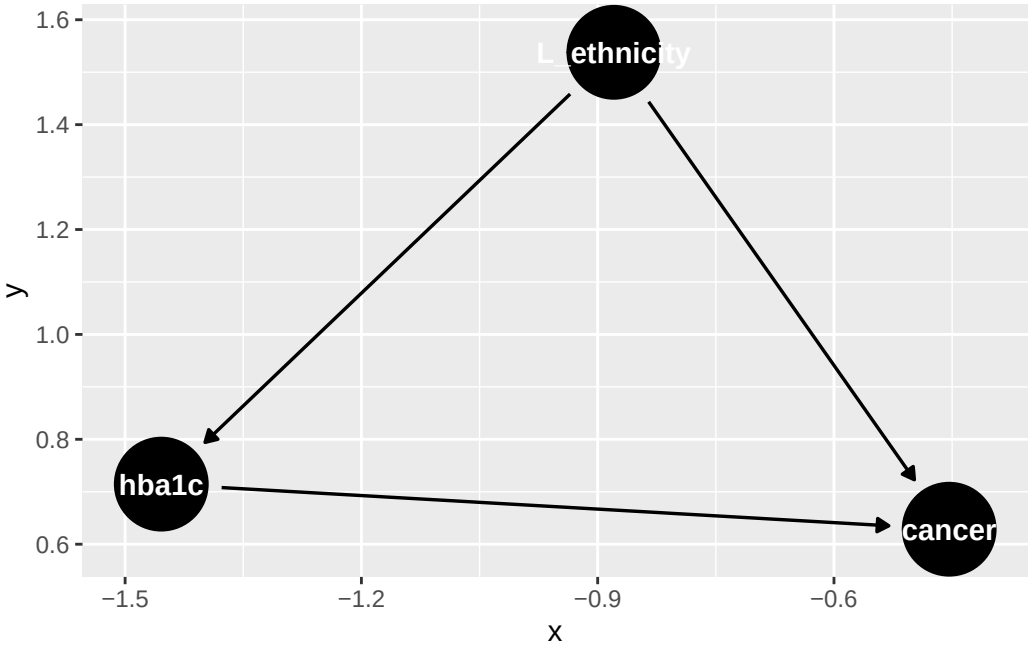
```
ggplot(popdatex, aes(x = hba1c, y = L_ethnicity, color = as.factor(cancer))) +
  geom_point() +
  geom_smooth(method = "lm", se = F) +
  theme_minimal() +
  labs(x = "hba1c",
       y = "Ethinicity",
       color = "cancer")
```

```
`geom_smooth()` using formula = 'y ~ x'
```



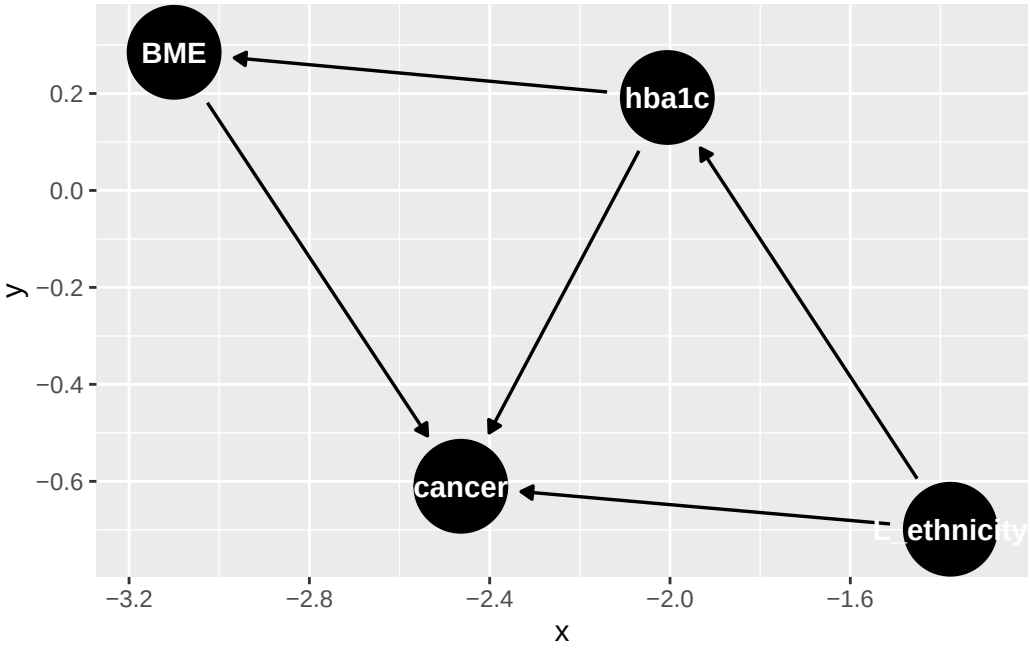
- The Direct Acyclic Diagram (DAG) to highlight the causal relationship between variables of interest

```
library(ggdag) # To create our DAGs
cdag <- ggdag::dagify(cancer ~ hba1c,
                      hba1c ~ L_ethnicity,
                      cancer ~ L_ethnicity)

ggdag::ggdag(cdag)
```



- Another possible relationship

```
library(ggdag) # To create our DAGs
moddag <- ggdag::dagify(cancer ~ BME,
                        cancer~ hba1c,
                            BME ~ hba1c,
                            hba1c ~ L_ethnicity,
                            cancer ~ L_ethnicity)

ggdag::ggdag(moddag)
```



Descriptive summary

```
suppressWarnings(suppressMessages(library(gtsummary)))

# summarize the data with our package
table1 <-
  popdatex |>
  tbl_summary(include = c(L_SEP, L_ethnicity, cancer, hba1c, BME, deprived))
table1
```

Characteristic	**N = 2,500**[1]
L_SEP	-0.01 (-0.69, 0.70)
L_ethnicity	-0.01 (-0.74, 0.68)
cancer	613 (25%)
hba1c	8.95 (7.95, 9.93)
BME	629 (25%)
deprived	
1	509 (20%)
2	512 (20%)

3	479 (19%)
4	488 (20%)
5	512 (20%)

[1]Median (Q1, Q3); n (%)

Causal modelling

Logistic regression

Unadjusted effect of hba1c

```
M0 <- glm(cancer ~ hba1c , family=binomial, data = popdatex)
t12 <- tbl_regression(M0, exponentiate=TRUE)
suppressMessages(t12)
```

Characteristic	OR[1]	95% CI[1]	p-value
hba1c	1.34	1.26, 1.43	<0.001

[1]OR = Odds Ratio, CI = Confidence Interval

```
# hba1c a significant cause for outcome - one could conclude assuming there all biases are a
```

```
# lets fit a model adjusting for
M1 <- glm(cancer ~ hba1c + as.factor(BME) + deprived + L_ethnicity,
family=binomial,
data = popdatex)
t2 <- tbl_regression(M1, exponentiate=TRUE)
suppressMessages(t2)
```

Characteristic	OR[1]	95% CI[1]	p-value
hba1c	1.06	0.98, 1.14	0.13
as.factor(BME)			

0	—	—	
1	1.42	1.03, 1.97	0.032
deprived	1.63	1.50, 1.77	<0.001
L_ethnicity	1.37	1.17, 1.60	<0.001

[1]OR = Odds Ratio, CI = Confidence Interval

Interpretation

The adjusted effect of hba1c is not statistically significant (aOR=1.06; 95% CI:0.98-1.14; p-value=0.13). This implies that a unit increase in hba1c, when adjusted for **bme, deprived, and ethnicity** results in a 6% increased risk for having cancer but this effect is not statistically significant. This maybe clinically significant though… it then becomes the decision of domain experts to further explore this effect and think of interventions if necessary.

- Try to predict cases of cancer within sample

```
# use model to predict cancer
popdatex$cancer_prob <- predict(M1, popdatex, type="response")
popdatex$c_pred = rep(0, dim(popdatex)[1])
popdatex$c_pred[popdatex$cancer_prob > .5] = 1
sumtable(round(popdatex, 1), out="return", digits=2)
```

```
      Variable    N   Mean Std. Dev.  Min Pctl. 25 Pctl. 75  Max
1        L_SEP 2500 -0.012         1 -3.7     -0.7      0.7  3.5
2  L_ethnicity 2500 -0.024         1 -3.4     -0.7      0.7  3.1
3       cancer 2500   0.25      0.43    0        0        0    1
4        hba1c 2500      9       1.5  3.7      7.9      9.9   15
5       sample 2500   0.75      0.43    0        1        1    1
6           id 2500   1250       722    1      626     1875 2500
7          BME 2500   0.25      0.43    0        0        1    1
8     deprived 2500      3       1.4    1        2        4    5
9  cancer_prob 2500   0.25      0.16    0      0.1      0.3  0.8
10      c_pred 2500  0.079      0.27    0        0        0    1
```

```
#create confusion matrix

CrossTable(popdatex$c_pred, popdatex$cancer)
```

```
   Cell Contents
|-------------------------|
|                       N |
| Chi-square contribution |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  2500


                | popdatex$cancer
popdatex$c_pred |         0 |         1 | Row Total |
----------------|-----------|-----------|-----------|
              0 |      1806 |       496 |      2302 |
                |     2.697 |     8.301 |           |
                |     0.785 |     0.215 |     0.921 |
                |     0.957 |     0.809 |           |
                |     0.722 |     0.198 |           |
----------------|-----------|-----------|-----------|
              1 |        81 |       117 |       198 |
                |    31.351 |    96.509 |           |
                |     0.409 |     0.591 |     0.079 |
                |     0.043 |     0.191 |           |
                |     0.032 |     0.047 |           |
----------------|-----------|-----------|-----------|
   Column Total |      1887 |       613 |      2500 |
                |     0.755 |     0.245 |           |
----------------|-----------|-----------|-----------|
```

- Only 19% of those with cancer are correctly predicted by the model, meaning it misses about 81% true cases. This model will not be useful as a risk scoring algorithm to use for prediction instances of cancer status.
- Now lets try and train the logistic model in the same way we would the classification algorithms (NB: logistic classifier is a logistic regression used for classification as in above example).

```r
# Encoding the target feature as factor and splitting data
# select 4 variables to include in the model
dcancer<-popdatex %>% select(c(cancer, hba1c, BME, deprived, L_ethnicity))
dcancer$cancer = factor(dcancer$cancer, levels = c(0, 1))
set.seed(123)
splitdat = sample.split(dcancer$cancer, SplitRatio = 0.70)

train = subset(dcancer, splitdat == TRUE)
test = subset(dcancer, splitdat == FALSE)
```

- Now lets train the logistic regression model (estimate f from data) so that we can use it to predict cancer status in new (test) data.

```r
train[-1] = scale(train[-1])
test[-1] = scale(test[-1])
M2 <- glm(cancer ~ hba1c + BME + deprived + L_ethnicity,
family=binomial,
data = train)
# use model to predict cancer
test$cancer_prob <- predict(M2, test, type="response")
test$c_pred = rep(0, dim(test)[1])
test$c_pred[test$cancer_prob > .5] = 1
#sumtable(round(popdatex, 1), out="return", digits=2)
#create confusion matrix
# library(gmodels)
CrossTable(test$c_pred, test$cancer)
```

```
   Cell Contents
|-------------------------|
|                       N |
| Chi-square contribution |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  750
```

```
             | test$cancer
 test$c_pred |         0 |         1 | Row Total |
-------------|-----------|-----------|-----------|
           0 |       547 |       155 |       702 |
             |     0.560 |     1.723 |           |
             |     0.779 |     0.221 |     0.936 |
             |     0.966 |     0.842 |           |
             |     0.729 |     0.207 |           |
-------------|-----------|-----------|-----------|
           1 |        19 |        29 |        48 |
             |     8.190 |    25.192 |           |
             |     0.396 |     0.604 |     0.064 |
             |     0.034 |     0.158 |           |
             |     0.025 |     0.039 |           |
-------------|-----------|-----------|-----------|
Column Total |       566 |       184 |       750 |
             |     0.755 |     0.245 |           |
-------------|-----------|-----------|-----------|
```

The test validation of the model shows that it even performs worse with data not seen managing to correctly identify cancer cases in 16% of those with disease.

Interpretation

- Now let's turn to algorithmic predictive models. These models are designed for prediction and not so much for inference, hence we will not have effect sizes, confidence intervals or p-values

Predictive modelling

Support vector machine example (linear)

```
# Feature Scaling
train[-1] = scale(train[-1])
test[-1] = scale(test[-1])
# Fitting SVM to the Training set
# install.packages('e1071')
```

```r
suppressWarnings(suppressMessages(library(e1071)))

classifier = svm(formula = cancer ~ .,
                 data = train,
                 type = 'C-classification',
                 kernel = 'linear', gamma=1)
classifier
```

```
Call:
svm(formula = cancer ~ ., data = train, type = "C-classification",
    kernel = "linear", gamma = 1)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1

Number of Support Vectors:  884
```

```r
# use the classifier function to predict new instances of cancer from data

# Predicting the Test set results
y_pred = predict(classifier, newdata = test[-1])
# y_pred
CrossTable(test$cancer,y_pred)
```

```
   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  750


             | y_pred
```

```
 test$cancer |         0 | Row Total |
-------------|-----------|-----------|
           0 |       566 |       566 |
             |     0.755 |           |
-------------|-----------|-----------|
           1 |       184 |       184 |
             |     0.245 |           |
-------------|-----------|-----------|
Column Total |       750 |       750 |
-------------|-----------|-----------|
```

Interpretation

Support vector machine (SVM) using linear kernel is not helpful, missing all those with cancer in the test dataset.

Support vector machine (Radial)

```
classrad = svm(formula = cancer ~ .,
                 data = train,
                 type = 'C-classification',
                 kernel = 'radial'
               #gamma = 1, cost = 1e5,
               )
# classifier
y_pred2 = predict(classrad, newdata = test[-1])
# y_pred
CrossTable(test$cancer,y_pred2)
```

```
   Cell Contents
|-------------------------|
|                       N |
| Chi-square contribution |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
```

```
|-------------------------|


Total Observations in Table:  750


             | y_pred2
 test$cancer |         0 |         1 | Row Total |
-------------|-----------|-----------|-----------|
           0 |       546 |        20 |       566 |
             |     0.408 |     6.237 |           |
             |     0.965 |     0.035 |     0.755 |
             |     0.776 |     0.435 |           |
             |     0.728 |     0.027 |           |
-------------|-----------|-----------|-----------|
           1 |       158 |        26 |       184 |
             |     1.254 |    19.186 |           |
             |     0.859 |     0.141 |     0.245 |
             |     0.224 |     0.565 |           |
             |     0.211 |     0.035 |           |
-------------|-----------|-----------|-----------|
Column Total |       704 |        46 |       750 |
             |     0.939 |     0.061 |           |
-------------|-----------|-----------|-----------|
```

Interpretation

With radial kernel, SVM does better and manages to separate those with and those without cancer showing 56% sensitivity. Accuracy also improves….

Support vector machine (polynomial)

```
classpoly = svm(formula = cancer ~ .,
                data = train,
                type = 'C-classification',
                kernel = 'polynomial',
               gamma = 1
               )
```

```r
# classifier
y_pred3 = predict(classpoly, newdata = test[-1])
# y_pred
CrossTable(test$cancer,y_pred3)
```

```
   Cell Contents
|-------------------------|
|                       N |
| Chi-square contribution |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  750


             | y_pred3
 test$cancer |         0 |         1 | Row Total |
-------------|-----------|-----------|-----------|
           0 |       548 |        18 |       566 |
             |     0.391 |     6.435 |           |
             |     0.968 |     0.032 |     0.755 |
             |     0.775 |     0.419 |           |
             |     0.731 |     0.024 |           |
-------------|-----------|-----------|-----------|
           1 |       159 |        25 |       184 |
             |     1.204 |    19.795 |           |
             |     0.864 |     0.136 |     0.245 |
             |     0.225 |     0.581 |           |
             |     0.212 |     0.033 |           |
-------------|-----------|-----------|-----------|
Column Total |       707 |        43 |       750 |
             |     0.943 |     0.057 |           |
-------------|-----------|-----------|-----------|
```

Interpretation

Same as radial, the polynomial kernel performs better

Random forest classifier

```
rfcancer <- randomForest(cancer~., data=train, proximity=TRUE)
print(rfcancer)
```

```
Call:
 randomForest(formula = cancer ~ ., data = train, proximity = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 23.43%
Confusion matrix:
     0  1 class.error
0 1244 77  0.05828917
1  333 96  0.77622378
```

Interpretation

classification error is high for those who have cancer but very low for non-cancer predictions.

Confusion matrix

```
# prediction within sample
p1 <- predict(rfcancer, train)
confusionMatrix(p1, train$cancer)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1303  147
```

```
         1   18  282

               Accuracy : 0.9057
                 95% CI : (0.891, 0.919)
    No Information Rate : 0.7549
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7165

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9864
            Specificity : 0.6573
         Pos Pred Value : 0.8986
         Neg Pred Value : 0.9400
             Prevalence : 0.7549
         Detection Rate : 0.7446
   Detection Prevalence : 0.8286
      Balanced Accuracy : 0.8219

       'Positive' Class : 0
```

```r
# prediction out of sample
p2 <- predict(rfcancer, test)
confusionMatrix(p2, test$cancer)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 525 144
         1  41  40

               Accuracy : 0.7533
                 95% CI : (0.7209, 0.7838)
    No Information Rate : 0.7547
    P-Value [Acc > NIR] : 0.5534

                  Kappa : 0.1787

 Mcnemar's Test P-Value : 6.421e-14
```

```
            Sensitivity : 0.9276
            Specificity : 0.2174
         Pos Pred Value : 0.7848
         Neg Pred Value : 0.4938
             Prevalence : 0.7547
         Detection Rate : 0.7000
   Detection Prevalence : 0.8920
      Balanced Accuracy : 0.5725

       'Positive' Class : 0
```

Sensitivity very high but specificity low. Accuracy much higher using RF.

XGBoost

```
# data dimension
# xgboost accepts numeric labels not factor
# Encoding the target feature as factor and splitting data
# select 4 variables to include in the model
dcancer<-popdatex %>% select(c(cancer, hba1c, BME, deprived, L_ethnicity))
dcancer$cancer = factor(dcancer$cancer, levels = c(0, 1))
set.seed(123)
splitdat = sample.split(dcancer$cancer, SplitRatio = 0.70)

traindat = subset(dcancer, splitdat == TRUE)
testdat = subset(dcancer, splitdat == FALSE)
traindat$cancer = as.numeric(traindat$cancer==2)
testdat$cancer = as.numeric(testdat$cancer==2)
class(traindat$cancer)
```

```
[1] "numeric"
```

```
dim(traindat[,-1])
```

```
[1] 1750    4
```

```
bstSparse <- xgboost(data = as.matrix(traindat[,-1]), label = traindat$cancer, max.depth = 2
```

```
[1] train-logloss:0.127473
[2] train-logloss:0.042988
```

```
# dense matrix
dtrain <- xgb.DMatrix(data = as.matrix(traindat[,-1]), label = traindat$cancer)
bstDMatrix <- xgboost(data = dtrain, max.depth = 2, eta = 1, nthread = 2, nrounds = 2, objec
```

```
[1] train-logloss:0.127473
[2] train-logloss:0.042988
```

```
# track learning progress
# verbose = 0, no message
bst <- xgboost(data = dtrain, max.depth = 2, eta = 1, nthread = 2, nrounds = 2, objective =

# verbose = 1, print evaluation metric
bst <- xgboost(data = dtrain, max.depth = 100, eta = 1, nthread = 2, nrounds = 10, objective
```

```
[1] train-logloss:0.127473
[2] train-logloss:0.042988
[3] train-logloss:0.015570
[4] train-logloss:0.005878
[5] train-logloss:0.002356
[6] train-logloss:0.001053
[7] train-logloss:0.000551
[8] train-logloss:0.000551
[9] train-logloss:0.000551
[10]    train-logloss:0.000551
```

```
# prediction
pred <- predict(bst, as.matrix(testdat[,-1]))
prediction <- as.numeric(pred > 0.5)
# size of the prediction vector
print(length(pred))
```

```
[1] 750
```

```
#CrossTable(testdat$cancer,prediction)
```

Fails to converge

Compare different algorithms

```
# prepare training scheme
control <- trainControl(method="repeatedcv", number=10, repeats=5)
# CART
set.seed(7)
fit.cart <- train(cancer~., data=dcancer, method="rpart", trControl=control)
# LDA
set.seed(7)
fit.lda <- train(cancer~., data=dcancer, method="lda", trControl=control)
# SVM
set.seed(7)
fit.svm <- train(cancer~., data=dcancer, method="svmRadial", trControl=control)
# kNN
set.seed(7)
#fit.knn <- train(dcancer~., data=dcancer, method="knn", trControl=control)
# Random Forest
set.seed(7)
fit.rf <- train(cancer~., data=dcancer, method="rf", trControl=control)
# collect resamples
results <- resamples(list(CART=fit.cart, LDA=fit.lda, SVM=fit.svm, RF=fit.rf))
```

Summary table

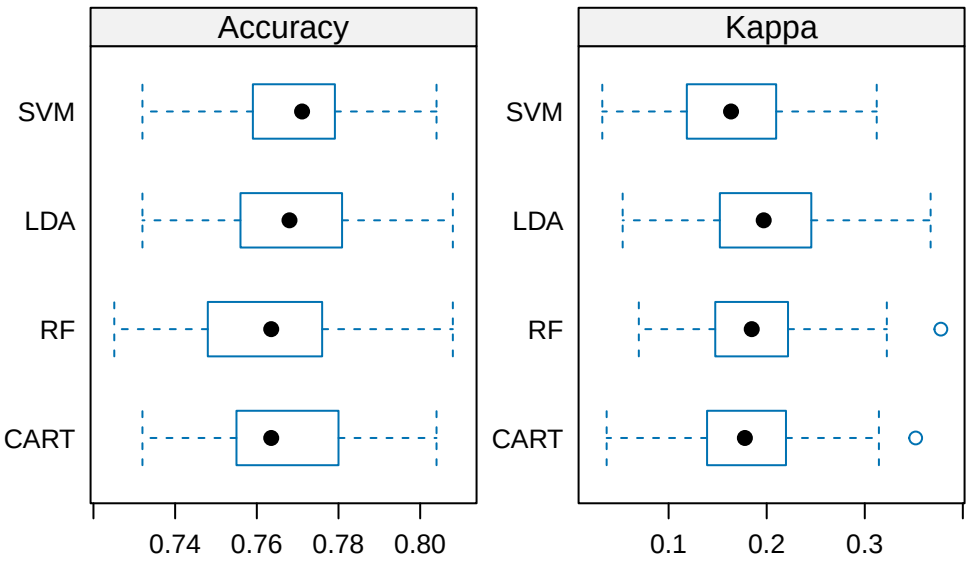
```
# summarize differences between modes
summary(results)
```

```
Call:
summary.resamples(object = results)

Models: CART, LDA, SVM, RF
Number of resamples: 50
```

```
Accuracy
          Min.    1st Qu.     Median       Mean   3rd Qu.  Max. NA's
CART 0.7320000 0.7550201 0.7635261 0.7648812 0.7797791 0.804    0
LDA  0.7320000 0.7562430 0.7680000 0.7684847 0.7806574 0.808    0
SVM  0.7320000 0.7592771 0.7710843 0.7678399 0.7783373 0.804    0
RF   0.7250996 0.7480000 0.7635261 0.7624041 0.7757751 0.808    0

Kappa
           Min.    1st Qu.     Median       Mean   3rd Qu.       Max. NA's
CART 0.03680276 0.1393694 0.1777547 0.1809116 0.2189202 0.3519204    0
LDA  0.05313899 0.1549939 0.1969489 0.1959878 0.2447596 0.3671554    0
SVM  0.03236287 0.1191784 0.1636622 0.1651889 0.2092980 0.3121842    0
RF   0.06961471 0.1476748 0.1847465 0.1873579 0.2211937 0.3775934    0
```

Box and Whisker

```
# box and whisker plots to compare models
scales <- list(x=list(relation="free"), y=list(relation="free"))
bwplot(results, scales=scales)
```



These comparisons show performance variations between different algorithms. There are other procedures that can be used to optimize the task in this example eg balancing classes, having as many variables as possible especially those that may be understood to influence condition of interest. The goal was to show that classical statistical modelling which encodes causal (inferential) explanation affords researchers to understand mechanisms of relationships between variables while predictive models are still ill equipped to help with inference. This is a very active research area in ML and data science. On the other hand, although statistical models can be used effectively to understand relationship mechanisms between variables, they are not suited to for optimizing prediction tasks.