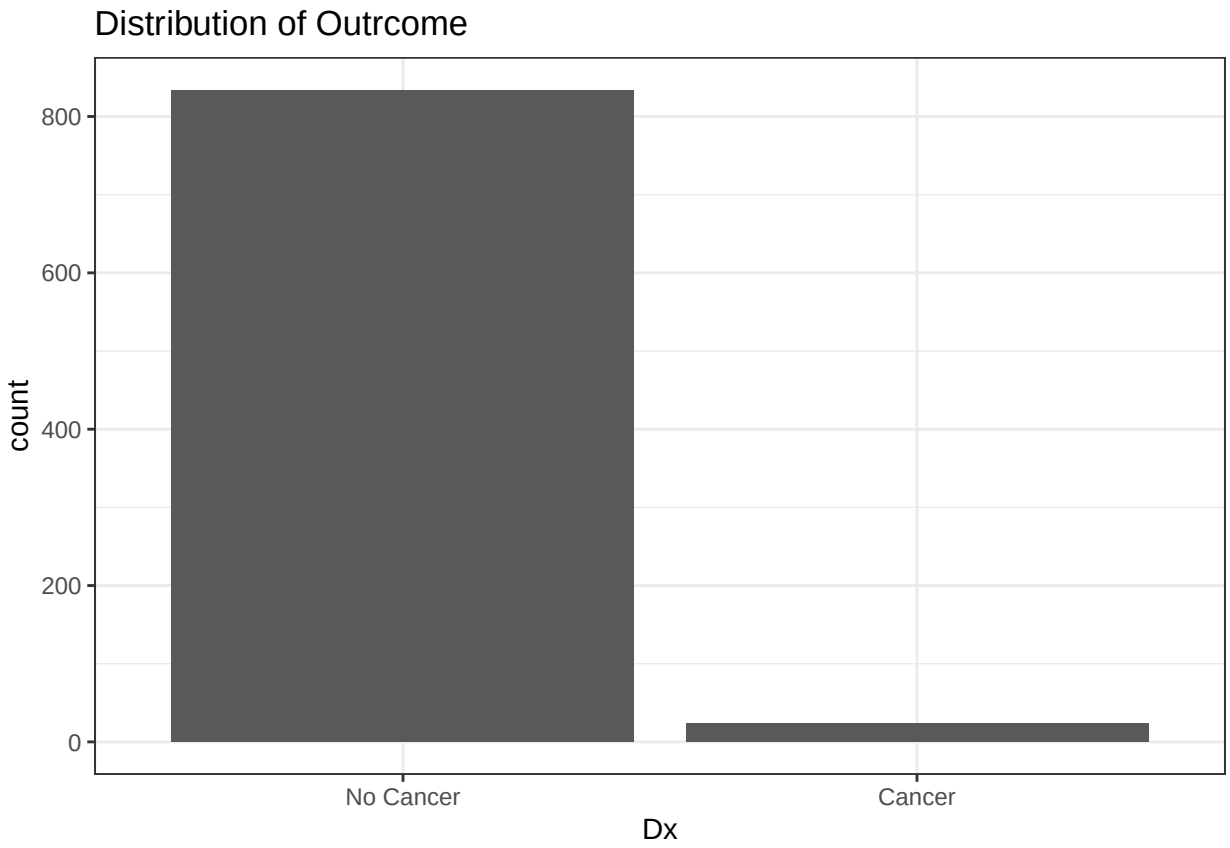
Final Project

Yupeng He & Martin Li

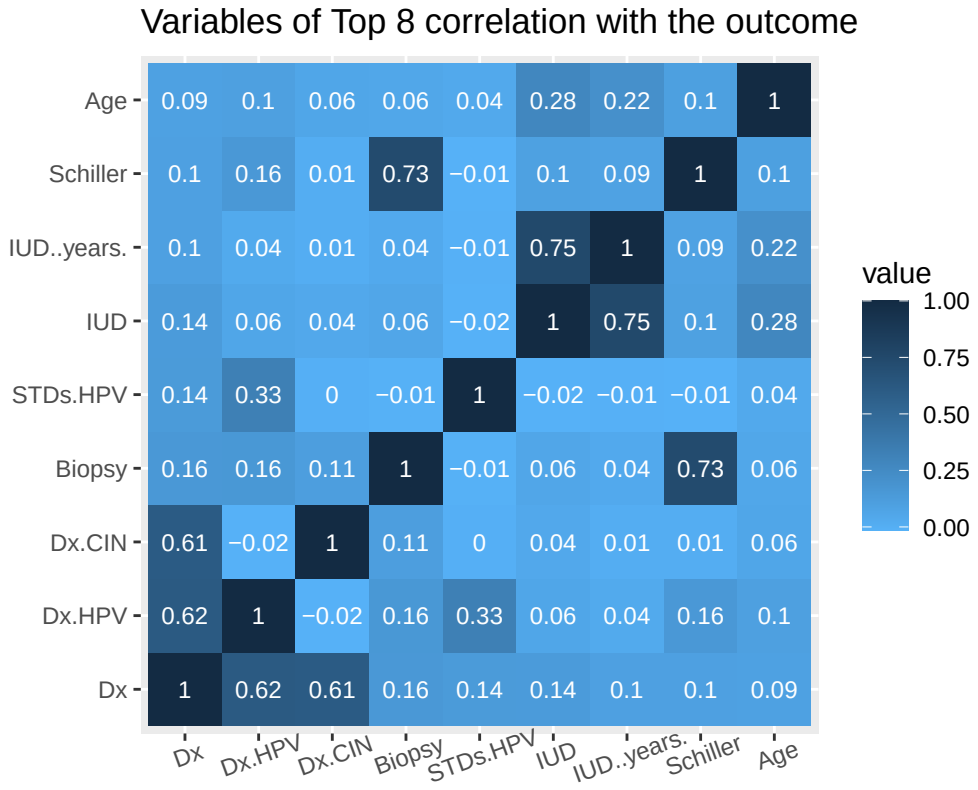
```
library(tidyverse)
library(dplyr)
library(caret)
library(e1071)
library(performanceEstimation)
library(UBL)
library(randomForest)
library(gbm)
library(ISLR)
library(ggplot2)
library(reshape)
library(pROC)
```

```
data <- read.csv("S:\\Downloads\\risk_factors_cervical_cancer.csv")
data[data == "?"] <- NA
# colSums(is.na(data))

cervical_cancer <- data

cervical_cancer <- as.data.frame(sapply(cervical_cancer, as.numeric))
cervical_cancer$Dx <- as.factor(cervical_cancer$Dx)

cervical_cancer <- cervical_cancer |>
  mutate(across(where(is.numeric), ~ coalesce(., median(., na.rm = TRUE))))

# factor_names <- c("Dx", "Dx.CIN",
#                   "Smokes", "Hormonal.Contraceptives", "IUD", "STDs",
#                   "STDs.condylomatosis", "STDs.vaginal.condylomatosis",
#                   "STDs.vulvo.perineal.condylomatosis", "STDs.syphilis",
#                   "STDs.pelvic.inflammatory.disease", "STDs.genital.herpes",
#                   "STDs.molluscum.contagiosum", "STDs.HIV", "STDs.Hepatitis.B",
#                   "STDs.cervical.condylomatosis", "STDs.AIDS",
#                   "Hinselmann", "Schiller", "Citology", "Biopsy")
#
# cervical_cancer[factor_names] <- lapply(cervical_cancer[factor_names] , factor)

cervical_cancer <- cervical_cancer |>
  select(-Dx.Cancer) |>
  mutate(across(where(is.numeric), scale))

cervical_cancer[is.na(cervical_cancer)] <- 0
cervical_cancer <- as.data.frame(sapply(cervical_cancer, as.numeric))
cor <- cor(cervical_cancer)
cervical_cancer$Dx <- as.factor(cervical_cancer$Dx)
```

```r
levels(cervical_cancer$Dx) <- c("No Cancer", "Cancer")
ggplot(data = cervical_cancer) +
  geom_bar(aes(x = Dx)) +
  ggtitle("Distribution of Outrcome") +
  theme_bw()
```




```r
top_name <- names(sort(abs(cor["Dx",]),decreasing = T)[1:9])
top_cor <- cor[top_name,top_name]
melt(top_cor) |>
  ggplot(aes(x = X1, y = X2, fill = value)) +
  geom_tile() +
  geom_text(aes(label = round(value,2)), color = "white", size = 3) +
  scale_fill_gradient(high = "#132B43", low = "#56B1F7") +
  coord_fixed() +
  ggtitle("Variables of Top 8 correlation with the outcome") +
  theme(axis.text.x = element_text(angle = 20)) +
  labs(x = "", y = "")
```



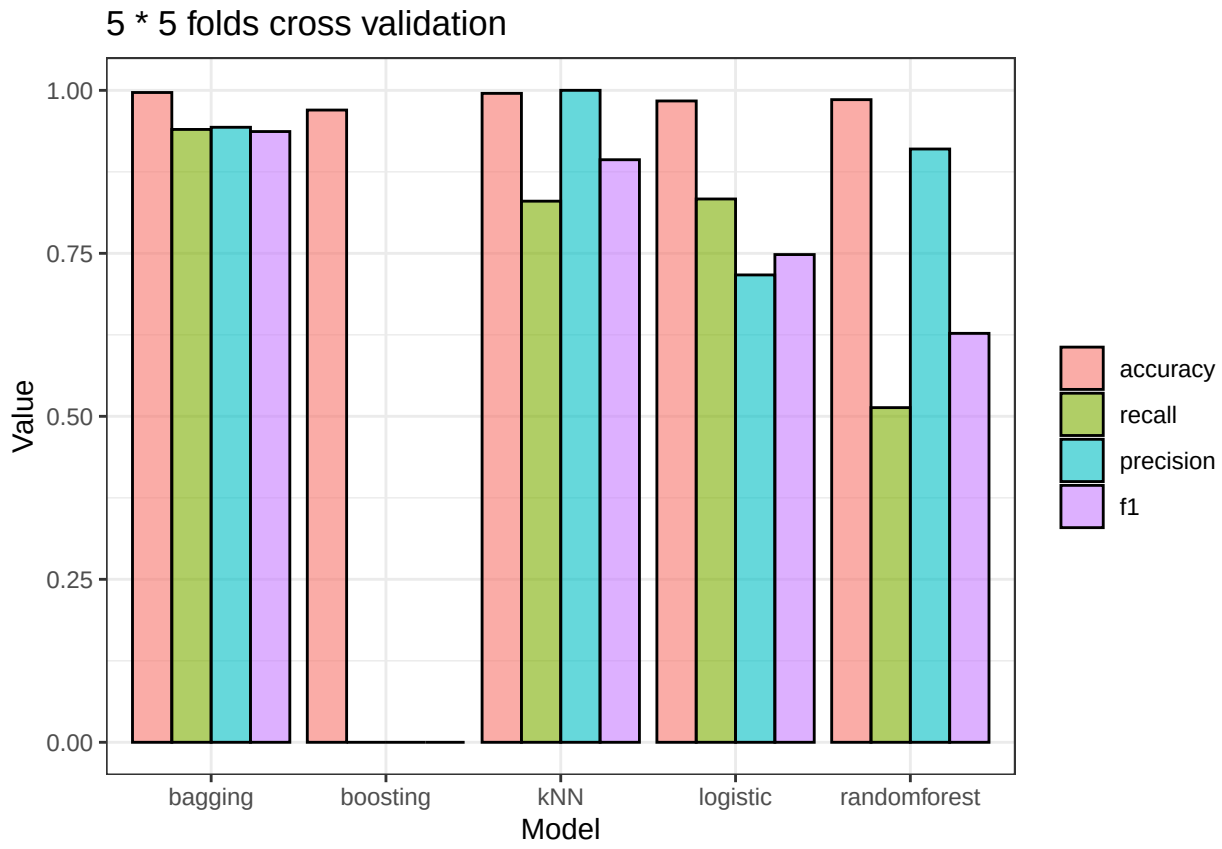

```r
# Stratify sampling
set.seed(1)
levels(cervical_cancer$Dx) <- c(1, 2)
train.index <- createDataPartition(cervical_cancer$Dx, p = 0.7, list = FALSE)
train <- cervical_cancer[ train.index,]
test  <- cervical_cancer[-train.index,]

# 5*5 CV
set.seed(1)
cv_index <- createMultiFolds(train$Dx, k = 5, times = 5)

cv_accuracy <- function(model_name){
  accuracy <- vector()
  recall <- vector()
  precision <- vector()
  f1 <- vector()

  for (fold in seq_along(names(cv_index))){
    fold_index <- unlist(cv_index[fold], use.names = F)

    cv_train <- train[ fold_index,]
    cv_test  <- train[-fold_index,]

    pred_label <- model_name(cv_train, cv_test)

    TP <- sum(pred_label == 2 & cv_test$Dx == 2)
```

```
    TN <- sum(pred_label == 1 & cv_test$Dx == 1)
    FP <- sum(pred_label == 2 & cv_test$Dx == 1)
    FN <- sum(pred_label == 1 & cv_test$Dx == 2)

    accuracy[fold] <- (TP+TN)/(TP+TN+FP+FN)
    recall[fold] <- ifelse(TP+FN==0, 0, TP/(TP+FN))
    precision[fold] <- ifelse(TP+FP==0, 0, TP/(TP+FP))
    f1[fold] <- 2*TP/(2*TP+FP+FN)
  }

  return(c(accuracy = mean(accuracy),
           recall = mean(recall),
           precision = mean(precision),
           f1 = mean(f1)))
}
```

```
logistic <- function(tr, te){
  model <- glm(Dx ~ ., data = tr, family = "binomial")
  pred_prob <- predict(model, newdata = te, type = "response")
  ifelse(pred_prob > 0.5, 2, 1)
}

knn <- function(tr, te){
  model <- knn3(Dx ~ ., data = tr, k = 5)
  predict(model, newdata = te, type = "class")
}

bagging <- function(tr, te){
  set.seed(1)
  p <- ncol(tr) - 1
  bag <- randomForest(Dx ~ .,
                      data = tr,
                      mtry = p,
                      importance = TRUE)
  predict(bag, newdata = te, type = "class")
}

rf <- function(tr, te){
  set.seed(1)
  rf <- randomForest(Dx ~ .,
                     data = tr,
                     importance = TRUE)
  predict(rf, newdata = te, type = "class")
}

boosting <- function(tr, te){
  set.seed(1)
  boost <- gbm(Dx ~ .,
               data = tr,
               distribution = "multinomial",
               n.trees = 5000, interaction.depth = 1, cv.folds = 5)

  min <- which.min(boost$cv.error)
```

```
  pred_prob <- as.data.frame(predict(boost, newdata = te, n.trees = min, type = "response"))[, 2]
  ifelse(pred_prob > 0.5, 2, 1)
}

cv <- sapply(list(logistic, knn, bagging, rf, boosting), cv_accuracy)
```

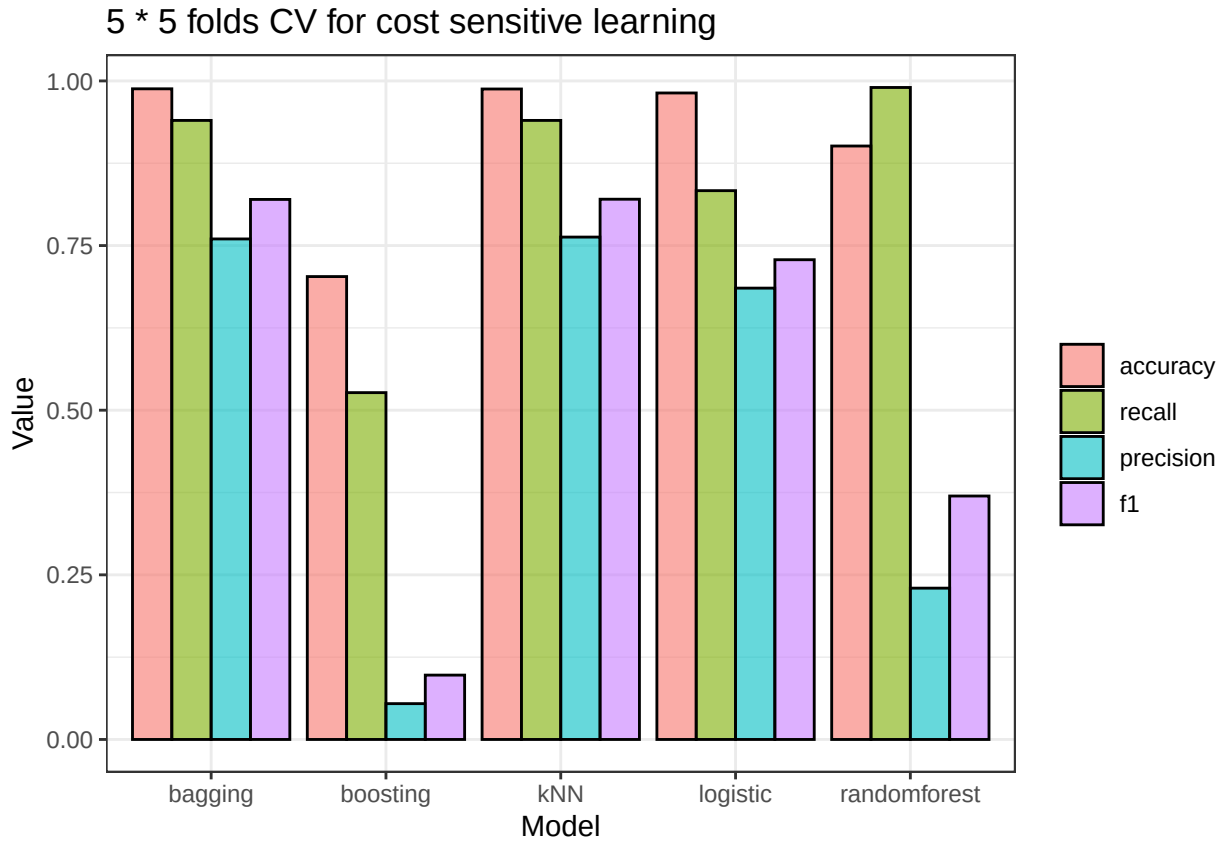
```
cv <- as.data.frame(cv)
names(cv) <- c("logistic", "kNN", "bagging", "randomforest", "boosting")
stats <- rownames(cv)
cv |>
  mutate(Stats = stats) |>
  pivot_longer(cols = 1:5,
               names_to = "Model",
               values_to = "Value") |>
  ggplot(aes(x = Model)) +
  geom_bar(aes(y = Value,
               fill = factor(Stats, levels = stats),
               group = factor(Stats, levels = stats)),
           stat = "identity",
           position = position_dodge(),
           color = "black",
           alpha = 0.6) +
  labs(fill = "") +
  ggtitle("5 * 5 folds cross validation") +
  theme_bw()
```



```r
# Cost sensitive learning
ap <- sum(cervical_cancer$Dx == 2)
an <- sum(cervical_cancer$Dx == 1)
threshold <- 1/(1+an/ap)

logistic_cs <- function(tr, te){
  model <- glm(Dx ~ ., data = tr, family = "binomial")
  pred_prob <- predict(model, newdata = te, type = "response")
  ifelse(pred_prob > threshold, 2, 1)
}

knn_cs <- function(tr, te){
  model <- knn3(Dx ~ ., data = tr, k = 5)
  pred_prob <- predict(model, newdata = te, type = "prob")[, 2]
  ifelse(pred_prob > threshold, 2, 1)
}

bagging_cs <- function(tr, te){
  set.seed(1)
  p <- ncol(tr) - 1
  bag <- randomForest(Dx ~ .,
                      data = tr,
                      mtry = p,
                      importance = TRUE)
  pred_prob <- predict(bag, newdata = te, type = "prob")[, 2]
  ifelse(pred_prob > threshold, 2, 1)
}

rf_cs <- function(tr, te){
  set.seed(1)
  rf <- randomForest(Dx ~ .,
                     data = tr,
                     importance = TRUE)
  pred_prob <- predict(rf, newdata = te, type = "prob")[, 2]
  ifelse(pred_prob > threshold, 2, 1)}

boosting_cs <- function(tr, te){
  set.seed(1)
  boost <- gbm(Dx ~ .,
               data = tr,
               distribution = "multinomial",
               n.trees = 5000, interaction.depth = 1, cv.folds = 5)

  min <- which.min(boost$cv.error)

  pred_prob <- as.data.frame(predict(boost, newdata = te, n.trees = min, type = "response"))[, 2]
  ifelse(pred_prob > threshold, 2, 1)
}

cs <- sapply(list(logistic_cs, knn_cs, bagging_cs, rf_cs, boosting_cs), cv_accuracy)
```

```r
cs <- as.data.frame(cs)
names(cs) <- c("logistic", "kNN", "bagging", "randomforest", "boosting")
```

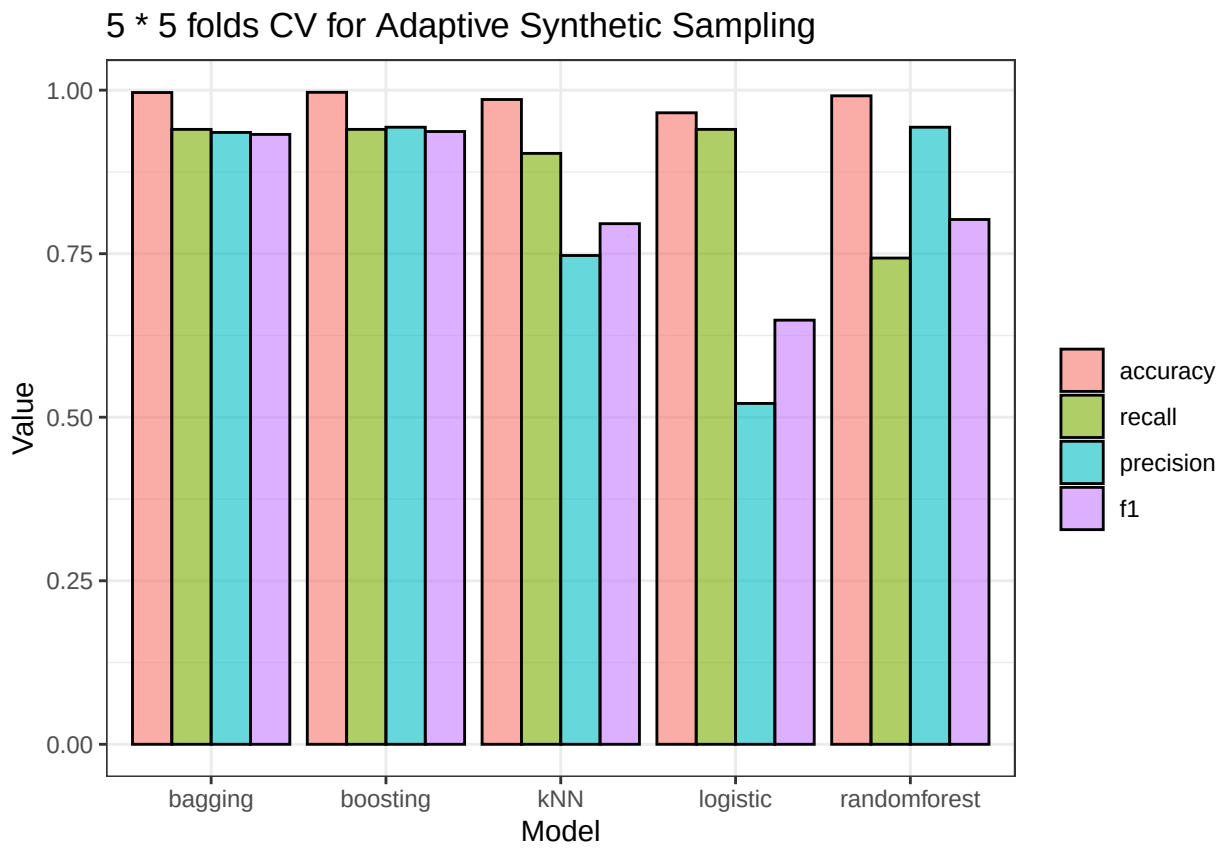
```r
stats <- rownames(cs)
cs |>
  mutate(Stats = stats) |>
  pivot_longer(cols = 1:5,
               names_to = "Model",
               values_to = "Value") |>
  ggplot(aes(x = Model)) +
  geom_bar(aes(y = Value,
               fill = factor(Stats, levels = stats),
               group = factor(Stats, levels = stats)),
           stat = "identity",
           position = position_dodge(),
           color = "black",
           alpha = 0.6) +
  labs(fill = "") +
  ggtitle("5 * 5 folds CV for cost sensitive learning") +
  theme_bw()
```



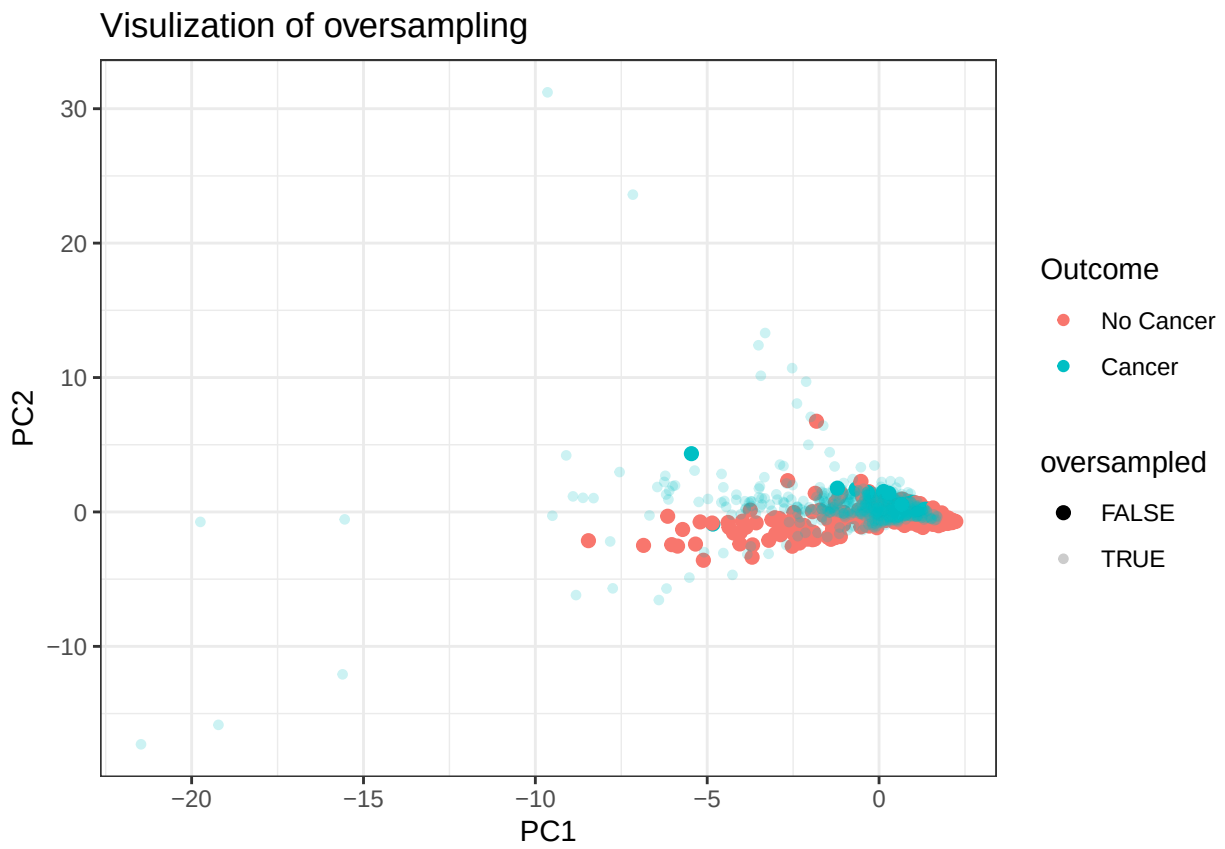

```r
# Adasyn
os_accuracy <- function(model_name){
  accuracy <- vector()
  recall <- vector()
  precision <- vector()
  f1 <- vector()
```

```
  for (fold in seq_along(names(cv_index))){
    fold_index <- unlist(cv_index[fold], use.names = F)

    cv_train <- AdasynClassif(Dx~., train[ fold_index,], beta = 1, k = 3)
    cv_test  <- train[-fold_index,]

    pred_label <- model_name(cv_train, cv_test)

    TP <- sum(pred_label == 2 & cv_test$Dx == 2)
    TN <- sum(pred_label == 1 & cv_test$Dx == 1)
    FP <- sum(pred_label == 2 & cv_test$Dx == 1)
    FN <- sum(pred_label == 1 & cv_test$Dx == 2)

    accuracy[fold] <- (TP+TN)/(TP+TN+FP+FN)
    recall[fold] <- ifelse(TP+FN==0, 0, TP/(TP+FN))
    precision[fold] <- ifelse(TP+FP==0, 0, TP/(TP+FP))
    f1[fold] <- 2*TP/(2*TP+FP+FN)
  }

  return(c(accuracy = mean(accuracy),
           recall = mean(recall),
           precision = mean(precision),
           f1 = mean(f1)))
}

Adasyn <- sapply(list(logistic, knn, bagging, rf, boosting), os_accuracy)
```

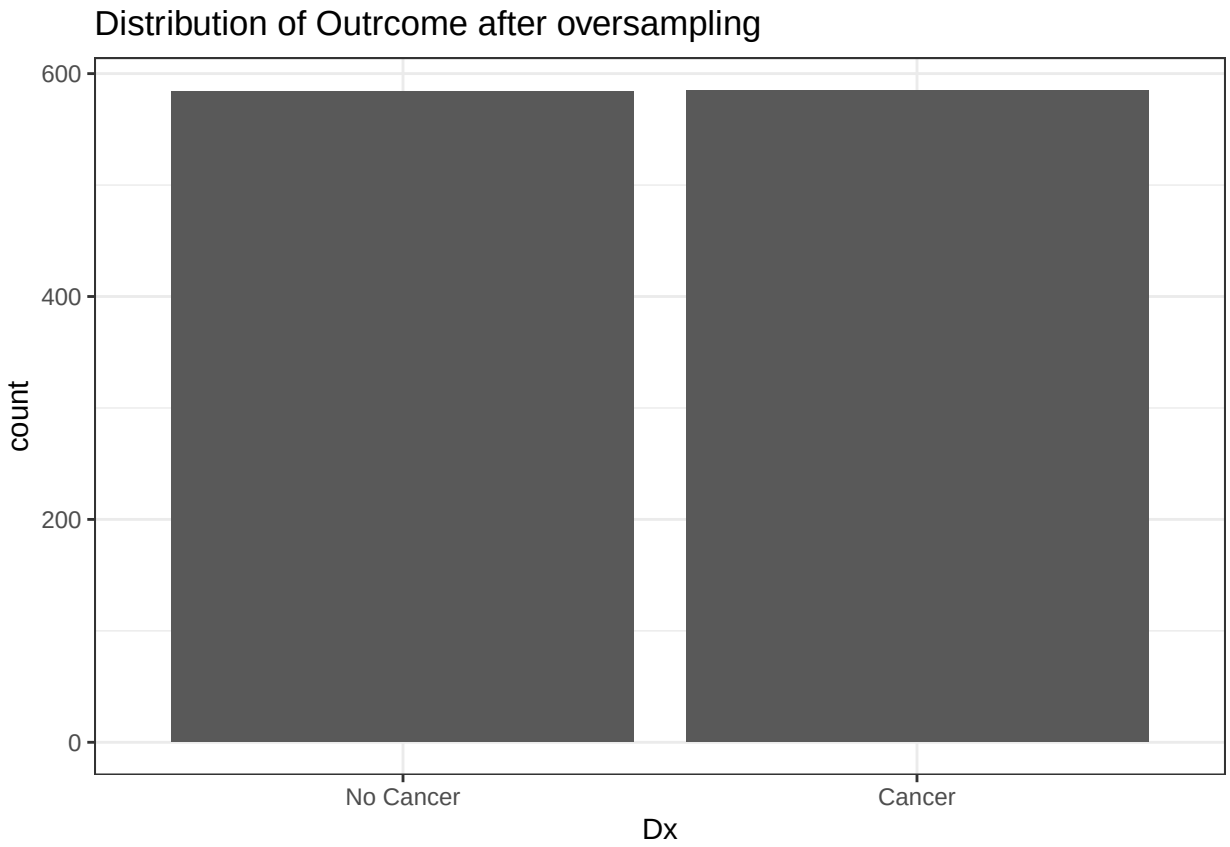
```
Adasyn <- as.data.frame(Adasyn)
names(Adasyn) <- c("logistic", "kNN", "bagging", "randomforest", "boosting")
stats <- rownames(Adasyn)
Adasyn |>
  mutate(Stats = stats) |>
  pivot_longer(cols = 1:5,
               names_to = "Model",
               values_to = "Value") |>
  ggplot(aes(x = Model)) +
  geom_bar(aes(y = Value,
               fill = factor(Stats, levels = stats),
               group = factor(Stats, levels = stats)),
           stat = "identity",
           position = position_dodge(),
           color = "black",
           alpha = 0.6) +
  labs(fill = "") +
  ggtitle("5 * 5 folds CV for Adaptive Synthetic Sampling") +
  theme_bw()
```




```
set.seed(1)
newdata <- AdasynClassif(Dx ~ ., train, beta = 1, k = 5)
pr.out <- prcomp(select(newdata,-Dx), scale = T)
os <- as.data.frame(pr.out$x[,1:2])
os$oversampled <- !do.call(paste0, newdata) %in% do.call(paste0, cervical_cancer)
os$Outcome <- newdata$Dx
levels(os$Outcome) <- c("No Cancer", "Cancer")
ggplot(os, aes(x = PC1,
               y = PC2, colour = Outcome,
               alpha = oversampled,
               size = oversampled,
               group = interaction(oversampled,Outcome))) +
  geom_point() +
  scale_alpha_manual(values=c(1,0.2)) +
  scale_size_manual(values=c(2,1.2)) +
  ggtitle("Visulization of oversampling") +
  theme_bw()
```



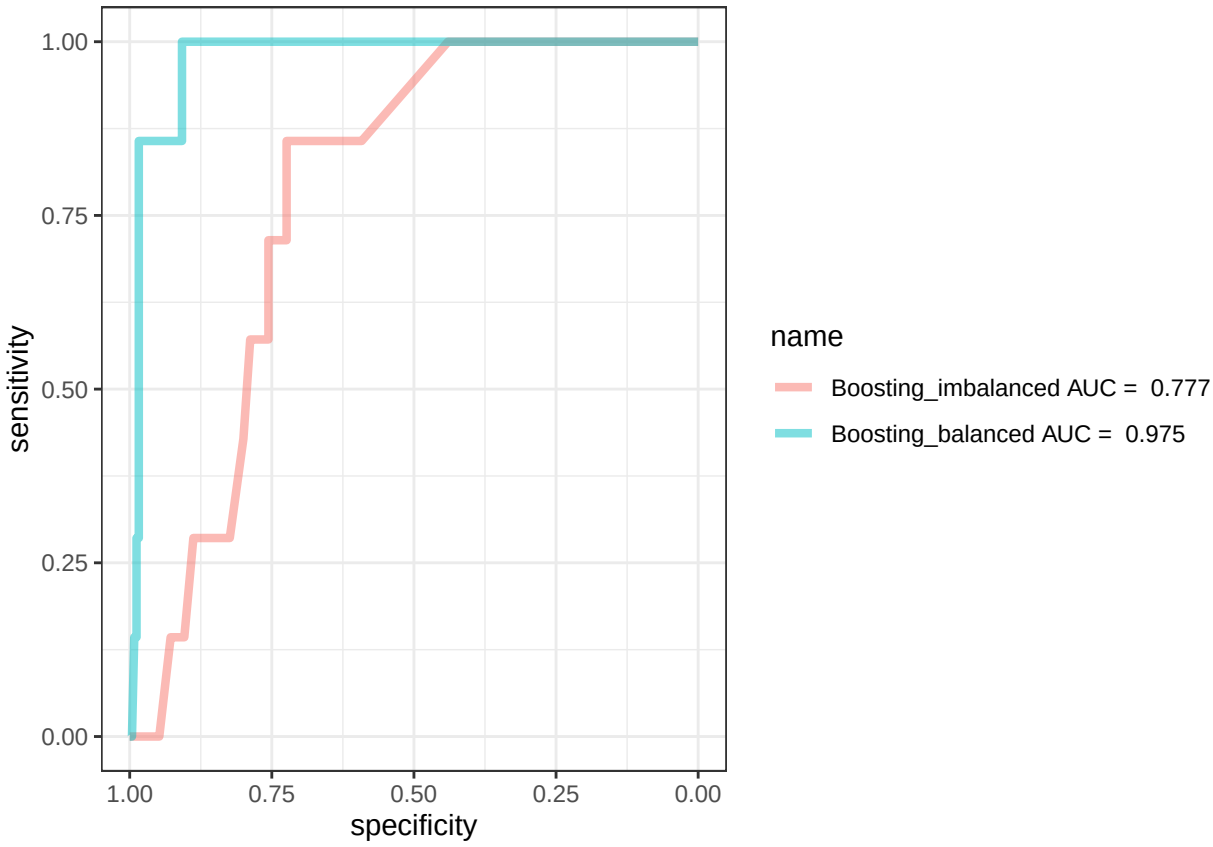

```
levels(newdata$Dx) <- c("No Cancer", "Cancer")
ggplot(data = newdata) +
  geom_bar(aes(x = Dx)) +
  ggtitle("Distribution of Outrcome after oversampling") +
  theme_bw()
```




```
levels(newdata$Dx) <- c(1,2)
```

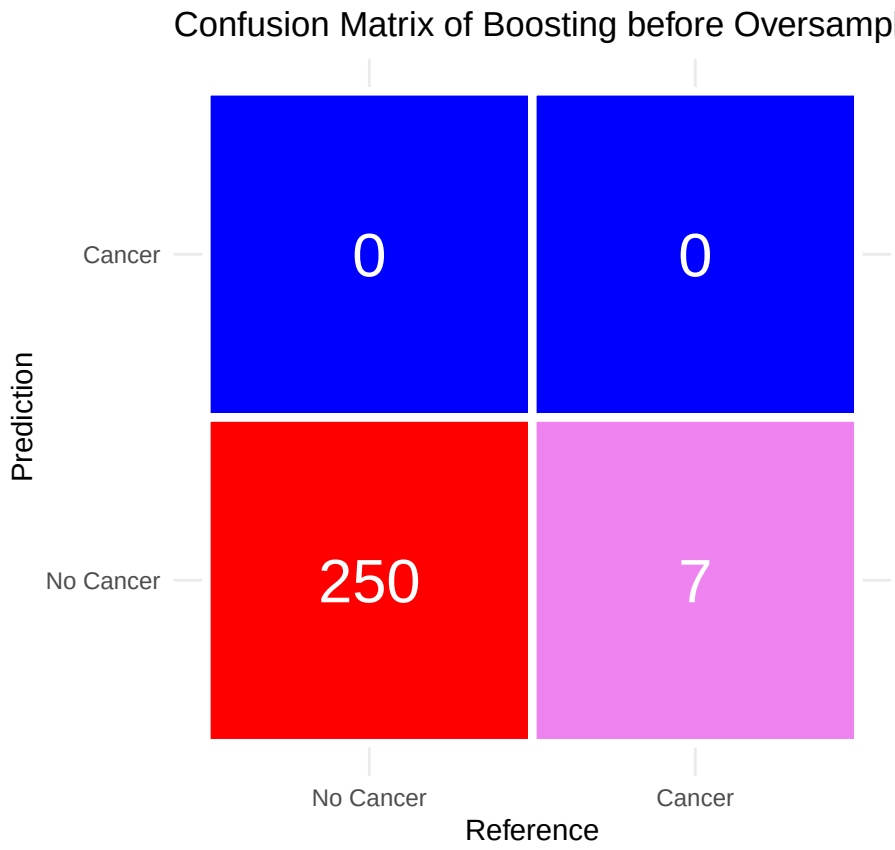
```
set.seed(1)
boost_1 <- gbm(Dx ~ .,
               data = train,
               distribution = "multinomial",
               n.trees = 5000, interaction.depth = 1, cv.folds = 5)
min1 <- which.min(boost_1$cv.error)

pred_prob <- as.data.frame(predict(boost_1, newdata = test, n.trees = min1, type = "response"))[, 2]
pred_label <- ifelse(pred_prob > 0.5, "Cancer", "No Cancer")
confusion1 <- confusionMatrix(data = factor(pred_label),
                              reference = factor(test$Dx, labels = c("No Cancer", "Cancer")))

roc1 <- roc(test$Dx, pred_prob)
auc1 <- auc(roc1)

set.seed(1)
boost_2 <- gbm(Dx ~ .,
               data = newdata,
               distribution = "multinomial",
               n.trees = 5000, interaction.depth = 1, cv.folds = 5)
min2 <- which.min(boost_2$cv.error)

pred_prob <- as.data.frame(predict(boost_2, newdata = test, n.trees = min2, type = "response"))[, 2]
pred_label <- ifelse(pred_prob > 0.5, "Cancer", "No Cancer")
```

```
confusion2 <- confusionMatrix(data = factor(pred_label),
                              reference = factor(test$Dx, labels = c("No Cancer", "Cancer")))

roc2 <- roc(test$Dx, pred_prob)
auc2 <- auc(roc2)

rocobjs <- list(Boosting_imbalanced = roc1, Boosting_balanced = roc2)
methods_auc <- paste(c("Boosting_imbalanced", "Boosting_balanced", "QDA","KNN"),
                     "AUC = ",
                     round(c(auc1, auc2),3))
ggroc(rocobjs, size = 1.5, alpha = 0.5) +
scale_color_discrete(labels = methods_auc) +
theme_bw()
```

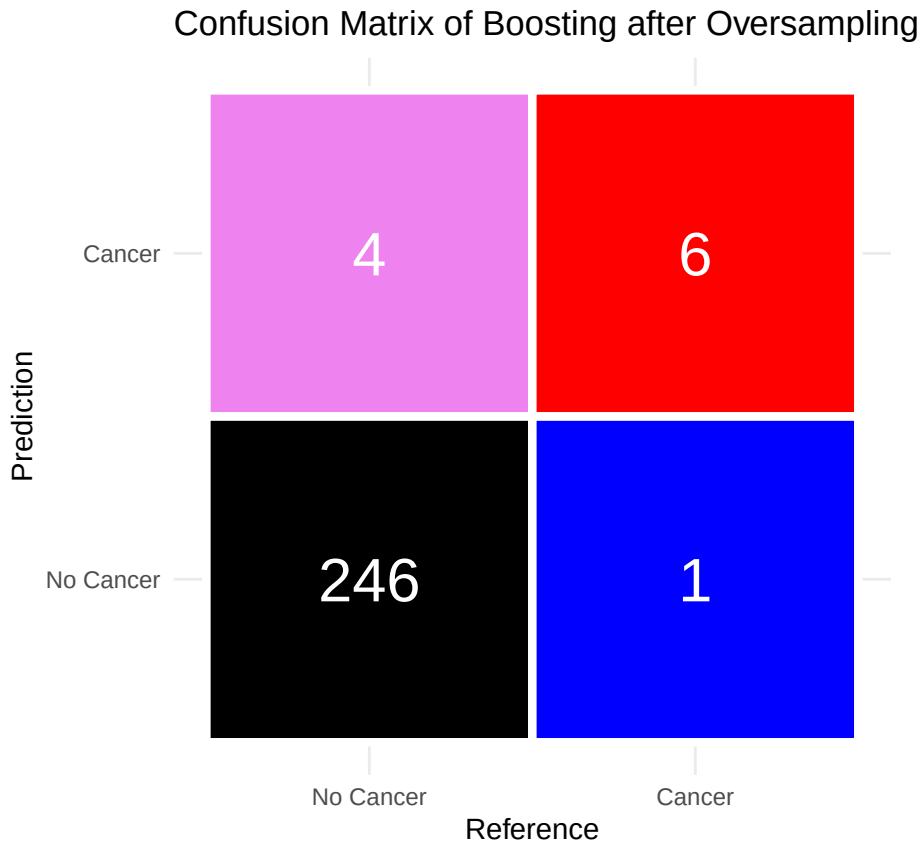



```
colors = c("blue", "violet","red", "black")

melt(confusion1$table) |>
  ggplot(aes(x = Reference, y = Prediction, fill = factor(value))) +
  geom_tile(color = "white",
            lwd = 1.5,
            linetype = 1) +
  geom_text(aes(label = value), color = "white", size = 8) +
  coord_fixed() +
  scale_fill_manual(values=colors) +
  ggtitle("Confusion Matrix of Boosting before Oversampling") +
```

```
  guides(fill = F) +
  theme_minimal()
```




```
melt(confusion2$table) |>
  ggplot(aes(x = Reference, y = Prediction, fill = factor(value))) +
  geom_tile(color = "white",
            lwd = 1.5,
            linetype = 1) +
  theme(legend.position = "none") +
  geom_text(aes(label = value), color = "white", size = 8) +
  coord_fixed() +
  scale_fill_manual(values=colors) +
  ggtitle("Confusion Matrix of Boosting after Oversampling") +
  guides(fill = F) +
  theme_minimal()
```




```
set.seed(1)

ctrl <- rfeControl(functions = treebagFuncs,
                   method = "repeatedcv",
                   repeats = 5,
                   verbose = F)

subsets <- c(1:5, 10, 15, 20, 25)

lmProfile <- rfe(x = select(newdata, -Dx),
                 y = newdata$Dx,
                 sizes = subsets,
                 rfeControl = ctrl)

rfe_name <- c("Dx", lmProfile$variables[1:8,2])
rfe <- newdata[, rfe_name]

set.seed(1)
boost_3 <- gbm(Dx ~ .,
             data = rfe,
             distribution = "multinomial",
             n.trees = 5000, interaction.depth = 1, cv.folds = 5)
min3 <- which.min(boost_3$cv.error)

pred_prob <- as.data.frame(predict(boost_3, newdata = test, n.trees = min3, type = "response"))[, 2]
pred_label <- ifelse(pred_prob > 0.5, "Cancer", "No Cancer")
```

```r
confusion3 <- confusionMatrix(data = factor(pred_label),
                              reference = factor(test$Dx, labels = c("No Cancer", "Cancer")))
```