**Machine learning code**

**Dataset**

```r
library(DMwR)
TACR <- knnImputation(TACRall)  # missing data imputation by KNN
n<-.03*nrow(TACR)              # split dataset
test.index<-sample(1:nrow(TACR),n)
TACR1<-TACR[-test.index,]
TACR2<-TACR[test.index,]
train <- TACR1  # training dataset
test <- TACR2   # validation dataset
```

**Decision tree**

```r
library(rpart)
train$SVR <- as.factor(train$SVR)
test$SVR <- as.factor(test$SVR)
fit <- rpart(SVR ~ ., data = train, control = rpart.control(minsplit = 3, cp = 0.001, maxdepth = 15))
pred <- predict(fit, test, type = "prob")
```

**Artificial neural networks**

```r
library(neuralnet)
nn <- neuralnet(SVR ~ ., train, hidden = 1, threshold = 0.01, stepmax = 1e+05, rep = 1, algorithm = "rprop+", linear.output = FALSE)
pred <- predict(nn, test)
```

**Random forest**

```r
library(randomForest)
train$SVR <- as.factor(train$SVR)
fit <- randomForest(train$SVR ~ ., data = train, ntree = 500, mtree = 7, nodesize = 1, maxnodes = NULL, importance = TRUE, proximity = TRUE)
pred <- predict(fit, type = "prob", test[,1:55])
```

**XGBoost**

```r
library(xgboost)
## binary classification
bst <- xgboost(data = as.matrix(train[, -56]), label = train$SVR, max_depth = 5, eta = 0.5, nthread = 2, nrounds = 200, objective = "binary:logistic")
pred <- predict(bst, as.matrix(test[, -56]))
```

```r
library(ggplot2)
## Plot feature importance as a bar graph
bst <- xgboost(data = as.matrix(TACRall[, -56]), label = TACRall$SVR, max_depth = 5, eta = 0.5, nthread = 2, nrounds = 200, objective = "binary:logistic")
```

importance_matrix <- xgb.importance(colnames(as.matrix(TACRall[, -56])), model = bst)
xgb.plot.importance(importance_matrix, rel_to_first = TRUE, xlab = "Relative importance")
gg <- xgb.ggplot.importance(importance_matrix, measure = "Frequency", rel_to_first = TRUE)
gg + ggplot2::ylab("Frequency")

## SHAP contribution dependency plots
xgb.plot.shap(as.matrix(TACRall[, -56]), model = bst, features = "logHCV_RNA")
contr <- predict(bst, as.matrix(TACRall[, -56]), predcontrib = TRUE)
xgb.plot.shap(as.matrix(TACRall[, -56]), contr, model = bst, top_n = 12, n_col = 3)

## SHAP summary plot
xgb.ggplot.shap.summary(as.matrix(TACRall[, -56]), contr, model = bst, top_n = 12)