***Code for the reviewer***

***## partial least regression***

*library(mixOmics)*

*x <- model.matrix( ~ variables, dataset)*

*y <- dataset$outcome*

*tune.splsda2 <- tune.splsda(x, y, validation = 'Mfold', folds = 10,*

*progressBar = FALSE, dist = 'max.dist',*

*nrepeat = 10)*

*tune.splsda2$choice.keepX*

*tune.splsda2$choice.ncomp$ncomp*

*choice.ncomp2 <- tune.splsda2$choice.ncomp$ncomp*

*choice.keepX2 <- tune.splsda2$choice.keepX[1:choice.ncomp2]*

*splsda.res2 <- splsda(x, y, ncomp = choice.ncomp2, keepX = choice.keepX2)*

*perf.splsda2 <- perf(splsda.res2, validation = "Mfold", folds = 10,*

*progressBar = FALSE, auc = TRUE, nrepeat = 10)*

*perf.splsda2$error.rate*

*selectVar(splsda.res2, comp = 1)$value*

*(S2<- mixOmics::plotLoadings(splsda.res2 , comp = 1,*

*contrib = 'max', method = 'mean',size.name = 2,*

*legend=F,title = '',*

*legend.color = color.mixo(2:1))*

*auc.plsda <- mixOmics::auroc(splsda.res2)*

***## Regularised random forest***

*data\_1 <-beat\_full %>% filter(Rx == "Belimumab")*

*data\_1 <-beat\_full %>% filter(Rx == "Placebo")*

*data\_2 <- data\_1 %>%*

*dplyr::select(variables,*

*major\_12)*

*sample <- createDataPartition(data\_2$major\_12, p = 0.6, list = F, times = 1)*

*data\_train <- data\_2[sample,]*

*data\_test <- data\_2[-sample,]*

*glimpse(data\_train)*

*glimpse(data\_test)*

*# train the dataset*

*#Evaluate the model with the default setting*

*#Find the best number of mtry = ?*

*#Find the best number of maxnodes*

*#Find the best number of ntrees = 10001*

*#Evaluate the model on the test dataset*

*library(randomForest)*

*library(caret)*

*library(e1071)*

*trControl <- trainControl(method = "cv",*

*number = 10,*

*search = "grid")*

*set.seed(1234)*

*# Run the model*

*rf\_default <- train(major\_12~.,*

*data = data\_train,*

*method = "RRF",*

*metric = "Accuracy",*

*ntree=10001,*

*tuneLenght = 10,*

*importance = TRUE,*

*trControl = trControl)*

*# Print the results*

*print(rf\_default)*

*set.seed(1234)*

*tuneGrid <- expand.grid(.mtry = c(1: 15))*

*rf\_mtry <- train(major\_12~.,*

*data = data\_train,*

*method = "RRF",*

*metric = "Accuracy",*

*ntree=10001,*

*tuneLenght = 10,*

*importance = TRUE,*

*trControl = trControl)*

*print(rf\_mtry)*

*# best m try*

*rf\_mtry$bestTune$mtry*

*max(rf\_mtry$results$Accuracy)*

*best\_mtry <- rf\_mtry$bestTune$mtry*

*best\_mtry*

*# maxnode size*

*store\_maxnode <- list()*

*tuneGrid <- expand.grid(.mtry = best\_mtry)*

*for (maxnodes in c(1: 50)) {*

*set.seed(1234)*

*rf\_maxnode <- train(major\_12~.,*

*data = data\_train,*

*method = "RRF",*

*metric = "Accuracy",*

*ntree=10001,*

*tuneLenght = 10,*

*importance = TRUE,*

*trControl = trControl,*

*maxnodes = maxnodes)*

*current\_iteration <- toString(maxnodes)*

*store\_maxnode[[current\_iteration]] <- rf\_maxnode*

*}*

*results\_mtry <- resamples(store\_maxnode)*

*summary(results\_mtry)*

*store\_maxtrees <- list()*

*for (ntree in c(201, 251, 301, 351, 401, 451, 501, 551, 601, 801, 1001, 2001,3001, 4001,5001,6001, 7001,8001,9001,10001,11001,12001,13001,14001,15001,16001,17001,18001,19001,20001)) {*

*set.seed(5678)*

*rf\_maxtrees <- train(major\_12~.,*

*data = data\_train,*

*method = "RRF",*

*metric = "Accuracy",*

*ntree=ntree,*

*tuneLenght = 10,*

*importance = TRUE,*

*trControl = trControl,*

*maxnodes = maxnodes)*

*key <- toString(ntree)*

*store\_maxtrees[[key]] <- rf\_maxtrees*

*}*

*results\_tree <- resamples(store\_maxtrees)*

*summary(results\_tree)*

*#### final model*

*fit\_rf <- train(major\_12~.,*

*data = data\_train,*

*method = "RRF",*

*metric = "Accuracy",*

*ntree=ntree,*

*tuneLenght = 10,*

*importance = TRUE,*

*trControl = trControl,*

*maxnodes = maxnodes)*

*# prediction*

*prediction <-predict(fit\_rf, data\_test)*

*# confusion matrix*

*confusionMatrix(prediction, data\_test$major\_12)*

*# variables*

*varImpPlot(fit\_rf)*

*varImpPlot(fit\_rf,n.var=min(10, nrow(fit\_rf$importance))) #bagging*

*varImpPlot(fit\_rf,n.var=min(10, nrow(fit\_rf$importance)),type = 2) #gini*

*varImpPlot(fit\_rf,n.var=min(10, nrow(fit\_rf$importance)),type = 1) #perm*