```
### RANDOM FOREST

#make a simple dataset to reduce to one entry per ground truth point (ID column and
class column)
sas_sub_data<-sas_data[,c(1,78)]
sas_sub_data$class<-as.factor(sas_sub_data$class)
sas_sub_data<-unique(sas_sub_data)

#Split dataset into training set (in this case 70%), here you need
#to adjust for splitting you want to do, depending on the class
#number you need to change the matrix size. And adjust the columns
#used for the random forest, in this example I only used the optical
#layers and not terrain for example.
sas_output<-list()
sas_all.scores<-c()
sas_c_matrix = matrix(0, 11, 11)
sas_all_cm<-c()

for(run.int in 1:30 ){
  run<-as.character(run.int)
  out_sas <- stratified(sas_sub_data, c("class"), 0.7)

  #create training dataset from the "out dataframe"
  sas_training.data<-sas_data[sas_data$ID %in% out_sas$ID,]

  #create validation dataset from the remaining data
  sas_validation.data<-sas_data[!sas_data$ID %in% out_sas$ID,]
  sas_validation.data<-sas_validation.data[,c(5:48,78)]

  #remove rows not to be included
  sas_training.data<-sas_training.data[,c(5:48,78)]
  sas_training.data$class<-as.factor(sas_training.data$class)

  #Run random forest model
  M_sas=rminer::fit(class~.,sas_training.data,model="randomForest", task = "class")
  sas_output[[run]]<-list(model=M_sas)
  #create dataframe whilst extracting the descriptor
  sas_table_res <- as.data.frame(M_sas@object$importance)

  #change order of values in column from highest to lowest
  sas_table_res_S <- sas_table_res[order(sas_table_res$MeanDecreaseAccuracy),]

  sas_output[[run]][["importance"]]<-sas_table_res_S

  # use the model to predict the classes in the validation dataset
  sas_validation.data$pred.class=rminer::predict(M_sas,sas_validation.data)

  sas_validation.data$class<-as.factor(sas_validation.data$class)

  # compare predicted versus observed data
  sas_c.m<-rminer::mmetric(sas_validation.data$class,
sas_validation.data$pred.class, metric=c("ALL"))
  sas_c.m1<-rminer::mmetric(sas_validation.data$class,
sas_validation.data$pred.class, metric=c("CONF"))

  # add all metrics to outputs and make table
  sas_output[[run]][["accuracy"]]<-sas_c.m
  sas_all.scores<-rbind(sas_all.scores, sas_c.m)

  # make a extra output for confusion matrix and total summed confusion matrix
  sas_output[[run]][["cfm"]]<-sas_c.m1$conf
  sas_c_matrix<-sas_c_matrix + sas_c.m1$conf

  # make a long table with all confusion matrices
  sas_all_cm<-rbind(sas_all_cm, sas_c.m1$conf)
}
dev.off()
```