# The Prediction of Asthma Attacks over a 12 week period

George

2023-07-23

## Decision Tree For Prediction Of Asthmatic Attacks

Asthma is a chronic lung disease affecting people of all ages.[1] It is caused by inflammation and muscle tightening around airways which has detrimental effects that can lead to death.[2] A supervised machine learning model known as Classification And Regression Tree (CART) which is a predictive model has been employed to explain how asthma variables values can be predicted based on other features. In order to evaluate the ability of the model to predict the outcome (Asthma Attack in 12 weeks) the data is partitioned to 70% (train) & 30% (test). A classification decision tree is built containing all the the features as the main model "fullmodel_train", with subsequent models with 20 features"f20", 10 features "f10" and 5 features "f5". These model variations as mentioned above would then be tested and assessed with their corresponding measures of performance eg. Sensitivity, PPV, NPV etc. to give better interpretation and prediction of asthmatic attacks based on the selected features. This would serve as a personalized risk assessment tool to assist primary care clinicians to predict asthmatic attacks over a period.

## Remove All Variables From Work space

## Load Librarys

```
library(tidyverse)
library(scales)
library(ranger)
library(ROSE)
library(pROC)
library(tibble)
library(rpart)
library(rpart.plot)
library(caret)
library(ggplot2)
library(dplyr)
library(tidyr)
```

## Assigning test_data and variables to conditions

```
test_data$CSA_3<-ifelse(test_data$CSA_3>4,4,test_data$CSA_3)
test_data$controllers<-ifelse(test_data$controllers>16,16,test_data$controllers)
test_data$reliever_use<-ifelse(test_data$reliever_use>4000,4000,test_data$reliever_use)
test_data$controllers<-ifelse(test_data$controllers>16,16, test_data$controllers)
```

## Vector for Range for each Variable. The range of the Variable can impact its contribution to the model. This is to control variables with a larger range from dominating the analysis and handle outliers.

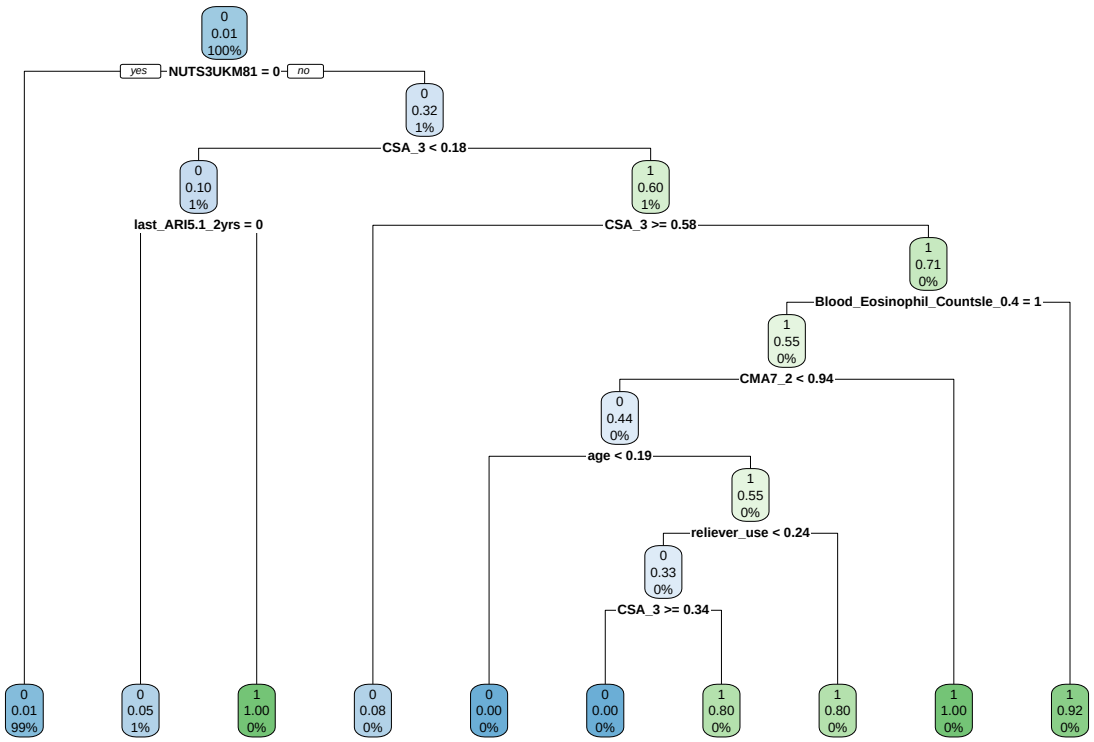
```
for(var in c("CSA_3","controllers","reliever_use","age")) {
  tempvar<-test_data[,var]
  range <- range(tempvar) # vector of range for each variable
  tempvar_s<-scale(tempvar, center = range[1], scale = range[2] - range[1])
  test_data[,var]<-tempvar_s
}
rm(tempvar_s,tempvar,var,range)
```

## Partition Data set & Train(70%) & Test(30%)

```
indexset <- sample(2,nrow(test_data), replace = T,prob = c(0.7,0.3))
train <- test_data[indexset==1,]
test <- test_data[indexset==2,]
```

## Build Classification tree for train set for all Variables

```
ctrl <- rpart.control(minsplit = 10, minbucket = 5, cp = 0.01)
train_tree_all_Var <- rpart(outcome ~ .,data=train, control = ctrl)
rpart.plot(train_tree_all_Var)
```




# Definition of parameters used in rpart.

minsplit:The minimum number of observations required to split a node. minbucket:The minimum number of observations in a terminal node (leaf) cp: complexity parameter, controls tree pruning.

## Decision Classification Tree of the Full Model.

The root node is NUTS3UKM81 which represents practice location. This is an important variable that was included in the decision tree because it provides valuable insights into the socioeconomic characteristics and disparities among different areas. Therefore, this feature allows the model to capture geographic variations and potential regional disparities in asthma prevalence, risk factors or healthcare access. This variable stands for the Nomenclature of Territorial Units for Statistics at level 3 which is the lowest and most detailed regional classification. The NUTS3UKM81 variable is the root node. However, it is observed that there is a branch at the first node which could lead to overfitting. This occurs where the model learns the training data too closely, capturing noise and random fluctuations rather than underlying patterns. If it says yes, the probability value of an asthmatic attack is 0.01 which is equivalent to 99% and if says no, the probability value of an asthmatic attack is 0.32 which is equivalent to 1%. The blue colored cells represent "NO" and the green colored represent "YES". After the root node, 8 branch nodes were extracted containing variables CSA_3,last_ARI5.1_2yrs=0,Blood_Eosinophil_Countsle_0.4,CMA7_2 <0.04,AGE < 0.19, reliever_use < 0.24. There are 10 leaf nodes at the end of the tree which have the lowest impurity.

## Variable Definitions:

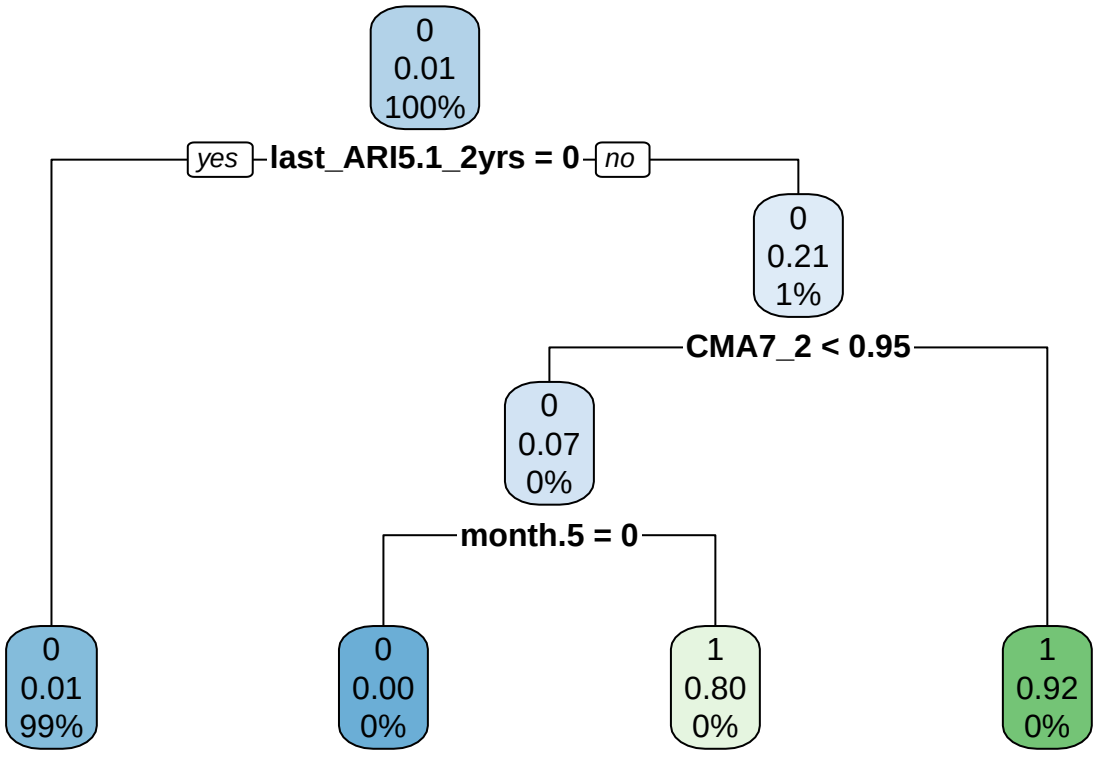
NUTS3UKM81 - Practice location CSA_3 - rolling avaerage of CSA over the last prescription last_ARI5.1_2yrs - Last Acute Respiratory Infection Blood_Eosinophil_Countsge_0.4 - Blood Eosinophil Count

## Full Model train without the NUTS variable.

```
NUTS_removal <- c("NUTS3")
matching_columns <- grep(paste(NUTS_removal, collapse = "|"), names(train), value = TRUE)
train2 <- train[, !(names(train) %in% matching_columns)]
```

## Classification tree without NUTS variable

```
train_all_Var_minusNUTS <- rpart(outcome ~ ., data=train2,control = ctrl)
rpart.plot(train_all_Var_minusNUTS)
```



## Predict train set for all Variables

```
pred_train_all_Var <- predict(train_tree_all_Var, train, type = "class")
```

## Prediction table for train set all Variables

```
table(pred_train_all_Var, train$outcome)
```

```
##
## pred_train_all_Var     0     1
##                  0 13880   163
##                  1     5    47
```

```
train_table_all_var <- table(pred_train_all_Var, train$outcome)
```

## Prediction test set for all variables

```
pred_test_all_Var <- predict(train_tree_all_Var, test, type = "class")
```

## Prediction table for test set all Variables

```
table(pred_test_all_Var, test$outcome)
```

```
##
## pred_test_all_Var    0    1
##                 0 5876   64
##                 1    5   24
```

```
test_table_all_var <- table(pred_test_all_Var, test$outcome)
```

## Predict train set without NUTS variable

```
pred_train_all_Var_minusNUTS <- predict(train_all_Var_minusNUTS, train, type = "class")
```

## Prediction table for train set without NUTS variable

```r
table(pred_train_all_Var_minusNUTS, train$outcome)
```

```
##
## pred_train_all_Var_minusNUTS     0     1
##                            0 13883   195
##                            1     2    15
```

```r
train_table_all_Var_minusNUTS <- table(pred_train_all_Var_minusNUTS, train$outcome)
```

## Predict test set without NUTS variable

```r
pred_test_all_Var_minusNUTS <- predict(train_all_Var_minusNUTS, test, type = "class")
```

## Prediction table for train set without NUTS variable

```r
table(pred_test_all_Var_minusNUTS, test$outcome)
```

```
##
## pred_test_all_Var_minusNUTS    0    1
##                            0 5876   83
##                            1    5    5
```

```r
test_table_all_Var_minusNUTS <- table(pred_test_all_Var_minusNUTS, test$outcome)
```

## SENSITIVITY, SPECIFICITY & PPV,NPV,F1,ACCURACY OF MODEL for train set All variables

```r
confusion_train <- confusionMatrix(train_table_all_var, mode = "everything", positive ="1")
```

## Convert confusion matrix of full model to data frame for train set / Rename Column

```r
df <- as.data.frame(confusion_train$byClass)
df <- tibble::rownames_to_column(df, var = "performance")
colnames(df)[2] <- "fullmodel_train"
```

## SENSITIVITY, SPECIFICITY & PPV,NPV,F1,ACCURACY OF MODEL for test_set All variables

```
confusion_test <- confusionMatrix(test_table_all_var, mode = "everything", positive ="1")
```

## Convert confusion matrix of full model to data frame on test set.

```
df_t <- as.data.frame(confusion_test$byClass)
df_t <- tibble::rownames_to_column(df_t, var = "performance")
colnames(df_t)[2] <- "fullmodel_test"
```

## SENSITIVITY, SPECIFICITY & PPV,NPV,F1,ACCURACY OF MODEL for train set without NUTS variable

```
confusion_train_minusNUTS <- confusionMatrix(train_table_all_Var_minusNUTS, mode = "everything", positi
```

## Convert confusion matrix of full model without NUTS to data frame on train set.

```
df_minusNUTS_train <- as.data.frame(confusion_train_minusNUTS$byClass)
df_minusNUTS_train <- tibble::rownames_to_column(df_minusNUTS_train, var = "performance")
colnames(df_minusNUTS_train)[2] <- "fullmodeltrain_minusNUTS"
```

## SENSITIVITY, SPECIFICITY & PPV,NPV,F1,ACCURACY OF MODEL for test set without NUTS variable

```
confusion_test_minusNUTS <- confusionMatrix(test_table_all_Var_minusNUTS, mode = "everything", positive
```

## Convert confusion matrix of full model without NUTS to data frame on test set.

```
df_minusNUTS_test <- as.data.frame(confusion_test_minusNUTS$byClass)
df_minusNUTS_test <- tibble::rownames_to_column(df_minusNUTS_test, var = "performance")
colnames(df_minusNUTS_test)[2] <- "fullmodeltest_minusNUTS"
```

## Merge the performance of the full model train & test, full model without NUTS train & test

```
merge_4models <- merge(merge(merge(df,df_t, by = "performance"),df_minusNUTS_train, by = "performance")
print(merge_4models)
```

```
##             performance fullmodel_train fullmodel_test fullmodeltrain_minusNUTS
## 1     Balanced Accuracy     0.611724711    0.635938539              0.535642266
## 2  Detection Prevalence     0.003689252    0.004858435              0.001206101
## 3        Detection Rate     0.003334516    0.004020774              0.001064207
## 4                    F1     0.358778626    0.410256410              0.132158590
## 5        Neg Pred Value     0.988392794    0.989225589              0.986148601
## 6        Pos Pred Value     0.903846154    0.827586207              0.882352941
## 7             Precision     0.903846154    0.827586207              0.882352941
## 8            Prevalence     0.014898900    0.014742838              0.014898900
## 9                Recall     0.223809524    0.272727273              0.071428571
## 10          Sensitivity     0.223809524    0.272727273              0.071428571
## 11          Specificity     0.999639899    0.999149804              0.999855960
##    fullmodeltest_minusNUTS
## 1             0.5279839931
## 2             0.0016753225
## 3             0.0008376612
## 4             0.1020408163
## 5             0.9860714885
## 6             0.5000000000
## 7             0.5000000000
## 8             0.0147428380
## 9             0.0568181818
## 10            0.0568181818
## 11            0.9991498045
```

## Build classification for important variables of the train_set

```
train_tree_all_Var$variable.importance
```

```
##                         CSA_3                     NUTS3UKM81
##                    34.0614776                     30.7086129
##                        CMA7_2 last_PC_attackgt_2yrs_unknown
##                    17.6693743                     13.8548302
##      recent_asthma_encounters                   reliever_use
##                    13.7402729                     11.1525108
##                         SIMD1               last_ARI5.1_2yrs
##                    10.9879973                     10.0116221
##       Nasal.SprayIn_Last_Year                            age
##                     9.6144976                      6.8153834
## Blood_Eosinophil_Countsge_0.4 Blood_Eosinophil_Countsle_0.4
##                     4.1474458                      4.1474458
##    Nasal.SprayIn_Last_5_Years                        month.2
```

```
##                    3.4305882                    2.6428754
##                     BTS_Step                     month.12
##                    2.5522743                    1.5151515
##         last_PC_attack1_3mon     last_ARI6.gt_2yrs_unknown
##                    1.4545455                    1.4351515
##                      month.5          last_PC_attack1_2yrs
##                    0.9503030                    0.7466667
##                      month.3
##                    0.4848485
```

```
Important_var <- train_tree_all_Var$variable.importance
Imp_var_df<- data.frame(Important_var)
head(Imp_var_df,20)
```

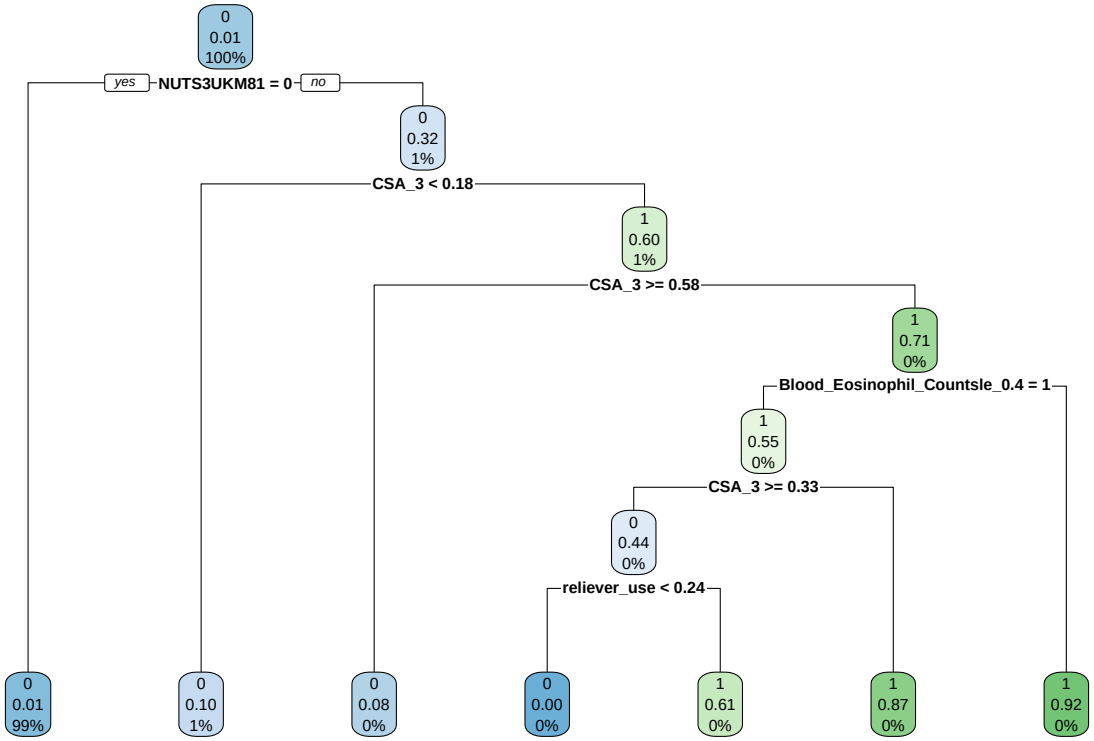
```
##                                Important_var
## CSA_3                           34.0614776
## NUTS3UKM81                      30.7086129
## CMA7_2                          17.6693743
## last_PC_attackgt_2yrs_unknown   13.8548302
## recent_asthma_encounters        13.7402729
## reliever_use                    11.1525108
## SIMD1                           10.9879973
## last_ARI5.1_2yrs                10.0116221
## Nasal.SprayIn_Last_Year          9.6144976
## age                              6.8153834
## Blood_Eosinophil_Countsge_0.4    4.1474458
## Blood_Eosinophil_Countsle_0.4    4.1474458
## Nasal.SprayIn_Last_5_Years       3.4305882
## month.2                          2.6428754
## BTS_Step                         2.5522743
## month.12                         1.5151515
## last_PC_attack1_3mon             1.4545455
## last_ARI6.gt_2yrs_unknown        1.4351515
## month.5                          0.9503030
## last_PC_attack1_2yrs             0.7466667
```

# BUILD MODEL TREE FOR TOP 20 FEATURES IN TRAIN MODEL

```
top_20_features_train <- rpart(outcome ~ CSA_3 + NUTS3UKM81 + CMA7_2 + last_PC_attackgt_2yrs_unknown + r
                               + last_ARI5.1_2yrs + Nasal.SprayIn_Last_Year + age + Blood_Eosinophil_Cou
                               + Nasal.SprayIn_Last_5_Years + month.2 + BTS_Step + month.12 + last_PC_at
                                data=train, method = "class")
```

## Decision tree for Top 20 features

```
rpart.plot(top_20_features_train)
```




# Prediction of Top 20 features train set

```
pred_top20_train <- predict(top_20_features_train, train, type = "class")
table(pred_top20_train, train$outcome)
```

```
##
## pred_top20_train     0     1
##                0 13875   168
##                1    10    42
```

```
train_table_20 <- table(pred_top20_train, train$outcome)
```

## Convert Top20_train into data.frame

```
df_top20_train <- data.frame(pred_top20_train, train$outcome)
```

## SENSITIVITY, SPECIFICITY & PPV,NPV,F1,ACCURACY OF TOP_20_ TRAIN MODEL

```
confusion_train_20 <- confusionMatrix(train_table_20, mode = "everything", positive = "1")

df2_train <- as.data.frame(confusion_train_20$byClass)
df2_train <- tibble::rownames_to_column(df2_train, var = "performance")
colnames(df2_train)[2] <- "f20 model_train"
```

## Merge both the performance of train set of full model & train set of top 20 features

```
merged_full_top20 <- merge(df, df2_train, by = "performance")
print(merged_full_top20)
```

```
##             performance fullmodel_train f20 model_train
## 1     Balanced Accuracy     0.611724711     0.599639899
## 2  Detection Prevalence     0.003689252     0.003689252
## 3        Detection Rate     0.003334516     0.002979780
## 4                    F1     0.358778626     0.320610687
## 5        Neg Pred Value     0.988392794     0.988036744
## 6        Pos Pred Value     0.903846154     0.807692308
## 7             Precision     0.903846154     0.807692308
## 8            Prevalence     0.014898900     0.014898900
## 9                Recall     0.223809524     0.200000000
## 10          Sensitivity     0.223809524     0.200000000
## 11          Specificity     0.999639899     0.999279798
```

## Comparing the performance metrics between the Fullmodel_train vs the F20model.

The sensitivity of the fullmodel_train is higher than the f20model which means it higher a true positive rate compared to f20model. In terms of predicting asthmatic attack the fullmodel_train can identify an attack better than the f20model if there is one. Also the fullmodel_train has better precision and positive predictive value which means that the likelihood for the fullmodel_train to predict a correct asthmatic attack when it occurs is higher than the f20model. The overall ability for the fullmodel_train to correctly classify both positive and negative attacks is better compared to the f20model. However, there was no significant difference in the specificity,negative predictive value and precision.

## Prediction & Contingency Table of Top 20 features in test set

```
pred_top20_test <- predict(top_20_features_train, test, type = "class")
table(pred_top20_test, test$outcome)
```

```
##
## pred_top20_test    0    1
##               0 5875   68
##               1    6   20
```

```
test_table_20 <- table(pred_top20_test, test$outcome)
```

## Top20_train into data.frame

```
df_top20_test <- data.frame(pred_top20_test, test$outcome)
```

## SENSITIVITY, SPECIFICITY & PPV,NPV,F1,ACCURACY OF TOP_20_ TEST MODEL

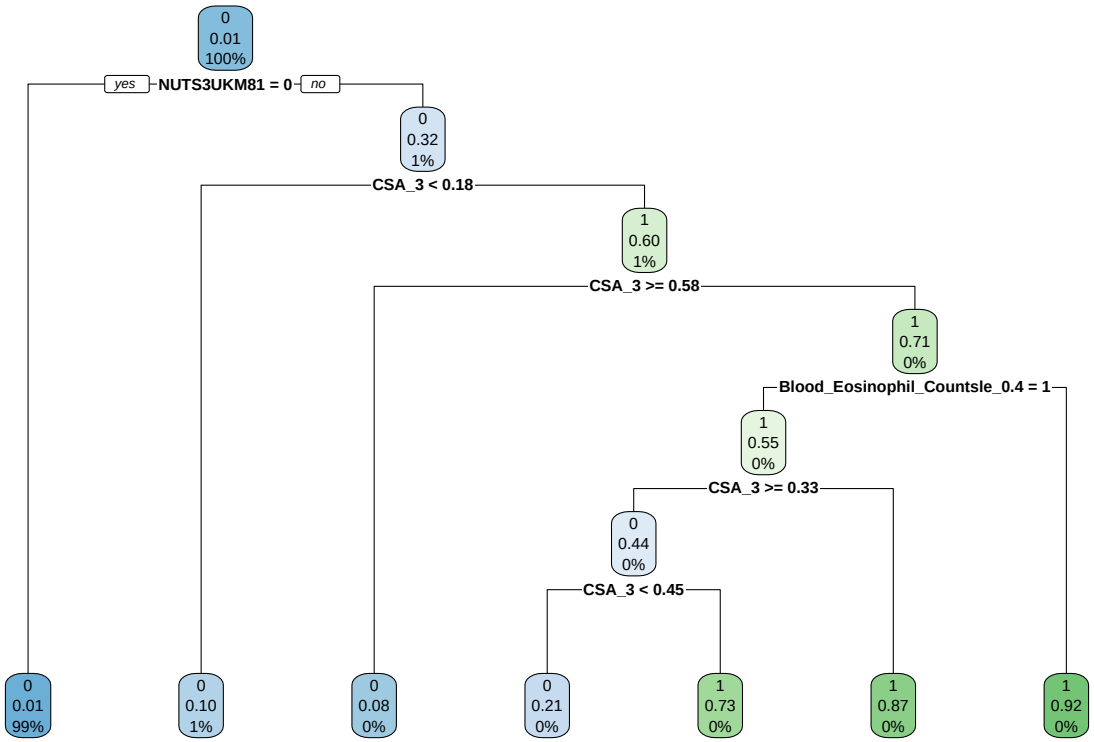
```
confusion_test_20 <- confusionMatrix(test_table_20, mode = "everything", positive = "1")
df2_test <- as.data.frame(confusion_test_20$byClass)
df2_test <- tibble::rownames_to_column(df2_test, var = "performance")
colnames(df2_test)[2] <- "f20 model_test"
```

## BUILD MODEL TREE FOR TOP 10 FEATURES IN TRAIN MODEL

```
top_10_features_train <- rpart(outcome ~ NUTS3UKM81 + Blood_Eosinophil_Countsge_0.4 + age +  Blood_Eosi
                    data=train, method = "class")
```

## Decision tree for Top 10 features

```
rpart.plot(top_10_features_train)
```




# Prediction of Top 10 features of train set

```
pred_top10_train <- predict(top_10_features_train, train, type = "class")
table(pred_top10_train, train$outcome)
```

```
##
## pred_top10_train     0     1
##                0 13879   171
##                1     6    39
```

```
train_table_10 <- table(pred_top10_train, train$outcome)
```

# Prediction of Top 10 features in test set

```
pred_top10_test <- predict(top_10_features_train, test, type = "class")
table(pred_top10_test, test$outcome)
```

```
##
## pred_top10_test    0    1
##               0 5878   70
##               1    3   18
```

```
test_table_10 <- table(pred_top10_test, test$outcome)
test_table_10
```

```
##
## pred_top10_test    0    1
##               0 5878   70
##               1    3   18
```

## Top10_train into data.frame

```
df_top10_train <- data.frame(pred_top10_train, train$outcome)
```

# SENSITIVITY, SPECIFICITY & PPV,NPV,F1,ACCURACY OF TOP_10_ TRAIN MODEL

```
confusion_train_10 <- confusionMatrix(train_table_10, mode = "everything", positive = "1")
```

## Convert confusion matrix of top 10 features into data frame

```
df3_train <- as.data.frame(confusion_train_10$byClass)
df3_train <- tibble::rownames_to_column(df3_train, var = "performance")
colnames(df3_train)[2] <- "f10 model_train"
```

# SENSITIVITY, SPECIFICITY & PPV,NPV,F1,ACCURACY OF TOP_10_ Test MODEL

```
confusion_test_10 <- confusionMatrix(test_table_10, mode = "everything", positive = "1")
```

## Convert confusion matrix of top 10 features into data frame

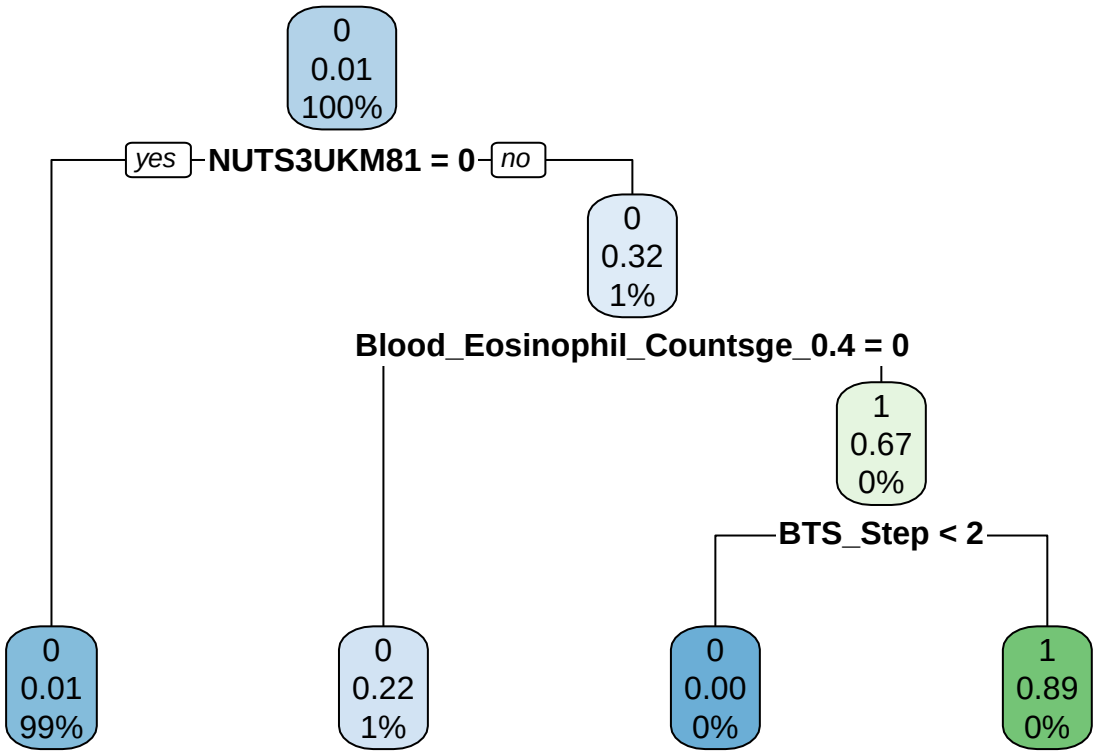
```
df3_test <- as.data.frame(confusion_test_10$byClass)
df3_test <- tibble::rownames_to_column(df3_test, var = "performance")
colnames(df3_test)[2] <- "f10 model_test"
```

# BUILD MODEL TREE FOR TOP 5 FEATURES IN TRAIN MODEL

```
top_5_features_train <- rpart(outcome ~ NUTS3UKM81 + Blood_Eosinophil_Countsge_0.4 + age +  Blood_Eosin
                              data=train, method = "class")
```

## Decision tree for Top 5 features features

```
rpart.plot(top_5_features_train)
```




# Prediction of Top 5 features of train set

```
pred_top5_train <- predict(top_5_features_train, train, type = "class")
table(pred_top5_train, train$outcome)
```

```
##
## pred_top5_train     0     1
##               0 13882   186
##               1     3    24
```

```
train_table_5 <- table(pred_top5_train, train$outcome)
```

## Prediction of Top 5 features in test set

```
pred_top5_test <- predict(top_5_features_train, test, type = "class")
table(pred_top5_test, test$outcome)
```

```
##
## pred_top5_test    0    1
##              0 5879   75
##              1    2   13
```

```
test_table_5 <- table(pred_top5_test, test$outcome)
test_table_5
```

```
##
## pred_top5_test    0    1
##              0 5879   75
##              1    2   13
```

## Top5_train into data.frame

```
df_top5_train <- data.frame(pred_top5_train, train$outcome)
```

## SENSITIVITY, SPECIFICITY & PPV,NPV,F1,ACCURACY OF TOP 5 train MODEL

```
confusion_train_5 <- confusionMatrix(train_table_5, mode = "everything", positive = "1")
```

## Convert confusion matrix of top 10 features into data frame

```
df4_train <- as.data.frame(confusion_train_5$byClass)
df4_train <- tibble::rownames_to_column(df4_train, var = "performance")
colnames(df4_train)[2] <- "f5 model_train"
```

## ENSITIVITY, SPECIFICITY & PPV,NPV,F1,ACCURACY OF TOP 5 test MODEL

```r
confusion_test_5 <- confusionMatrix(test_table_5, mode = "everything", positive = "1")
```

## Convert confusion matrix of top 10 features into data frame

```r
df4_test <- as.data.frame(confusion_test_5$byClass)
df4_test <- tibble::rownames_to_column(df4_test, var = "performance")
colnames(df4_test)[2] <- "f5 model_test"
```

## Merge all Models with performance

```r
merged_data <- df %>%
  left_join(df_t, by = "performance") %>%
  left_join(df_minusNUTS_train, by = "performance") %>%
  left_join(df_minusNUTS_test, by = "performance") %>%
  left_join(df2_train, by = "performance") %>%
  left_join(df2_test, by = "performance") %>%
  left_join(df3_train, by = "performance") %>%
  left_join(df3_test, by = "performance") %>%
  left_join(df4_train, by = "performance") %>%
  left_join(df4_test, by = "performance")
```

## Create the sensitivity against all models.

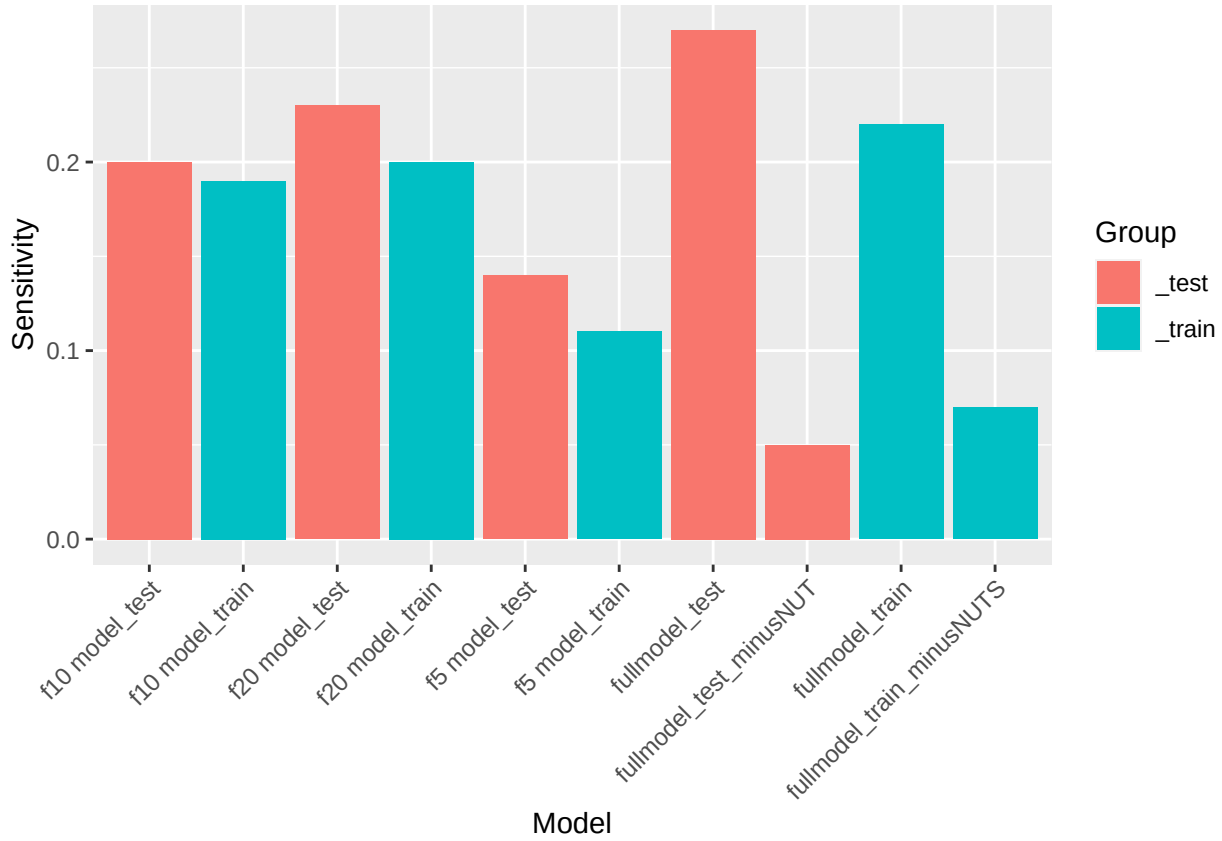
```r
sensitivity_data <- data.frame(
  Model = c("fullmodel_train","fullmodel_test","fullmodel_train_minusNUTS","fullmodel_test_minusNUT", "
  Sensitivity = c(0.22, 0.27, 0.07, 0.05, 0.20, 0.23, 0.19, 0.20, 0.11, 0.14 )
)
```

## Filter & reshape

```r
sensitivity_data <- sensitivity_data %>%
  mutate(Group = ifelse(grepl("_train", Model), "_train", "_test"))
```

## Plot Graph

```r
ggplot(sensitivity_data, aes(x = Model, y = Sensitivity, fill = Group)) +
  geom_bar(stat = "identity", position = "dodge") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(x = "Model", y = "Sensitivity", fill = "Group")
```




# Create the PPV against all models.

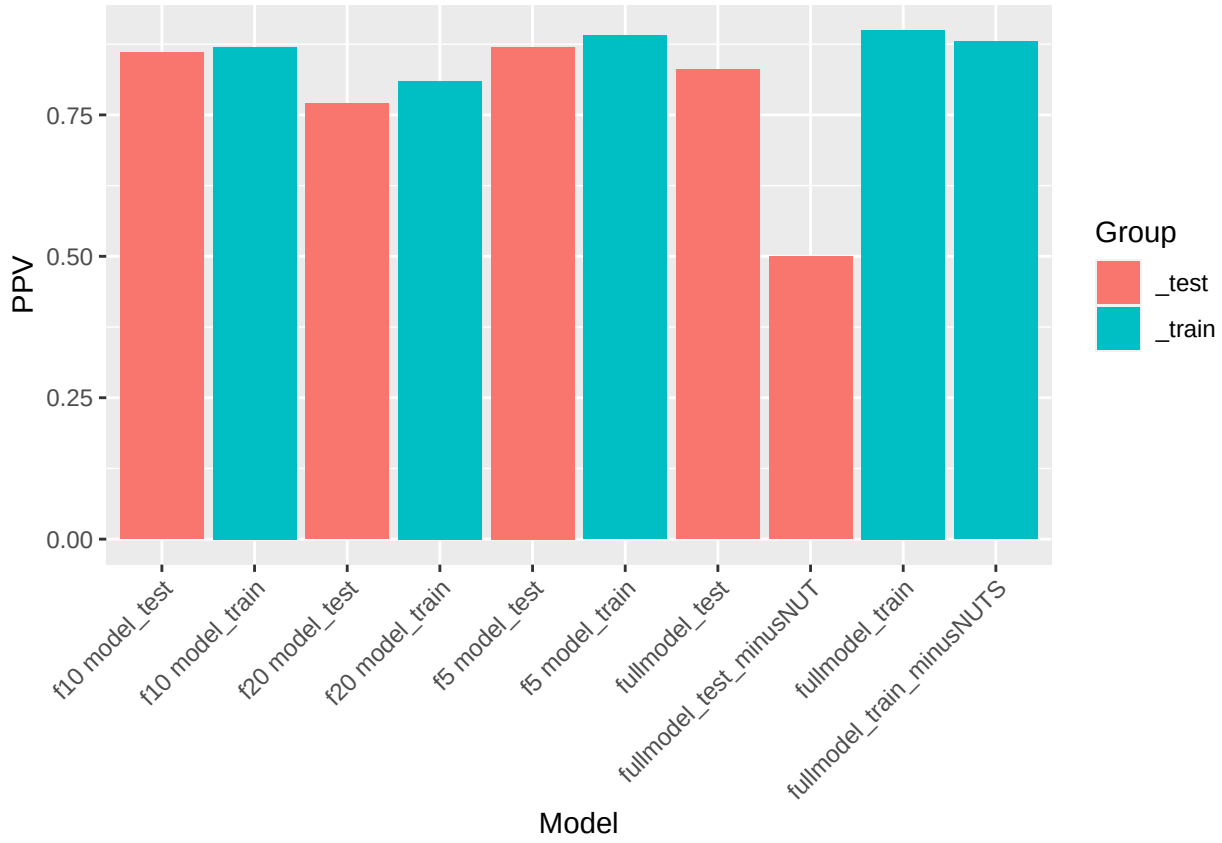
```
ppv_data <- data.frame(
  Model = c("fullmodel_train","fullmodel_test","fullmodel_train_minusNUTS","fullmodel_test_minusNUT", "
  PPV = c(0.90, 0.83, 0.88, 0.50, 0.81, 0.77, 0.87, 0.86, 0.89, 0.87 )
)
```

## Filter & reshape

```
ppv_data <- ppv_data %>%
  mutate(Group = ifelse(grepl("_train", Model), "_train", "_test"))
```

## Plot Graph

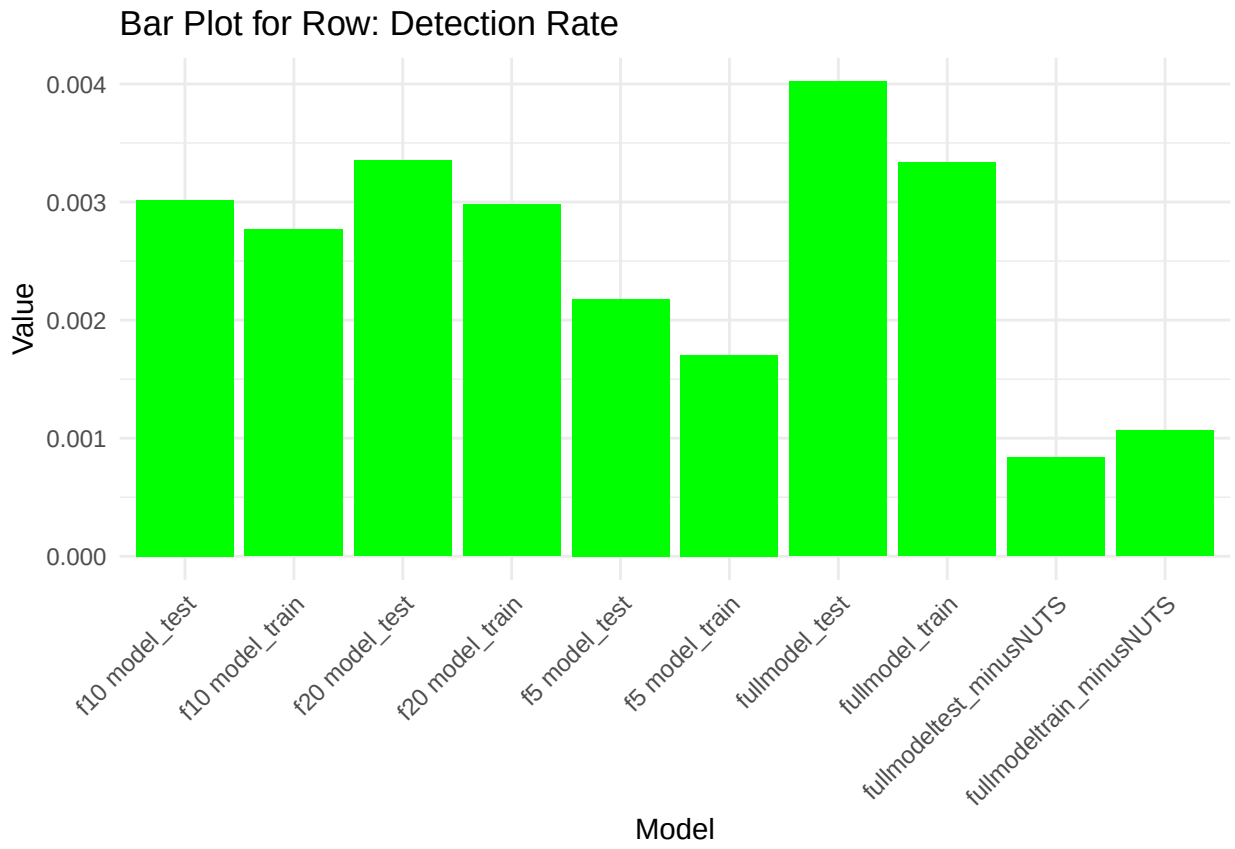
```
ggplot(ppv_data, aes(x = Model, y = PPV, fill = Group)) +
  geom_bar(stat = "identity", position = "dodge") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(x = "Model", y = "PPV", fill = "Group")
```




```
long_df <- pivot_longer(
  merged_data,
  cols = starts_with("f"),
  names_to = "Model",
  values_to = "Value"
)
```

## Graph of Performcnce for merged_data

```
row_to_plot <- "Detection Rate"
specific_row_data <- long_df %>% filter(performance == row_to_plot)
ggplot(specific_row_data, aes(x = Model, y = Value)) +
  geom_bar(stat = "identity", fill = "green") +
  labs(x = "Model", y = "Value", title = paste("Bar Plot for Row:", row_to_plot)) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




# References

1. Sin, D. D. *et al.* What is asthma-COPD overlap syndrome? Towards a consensus definition from a round table discussion. *Eur Respir J* **48**, 664–673 (2016).

2. Hargreave, F. E. & Nair, P. The definition and diagnosis of asthma. *Clin Exp Allergy* **39**, 1652–1658 (2009).