ML Project

Suzana Iacob

21/11/2019

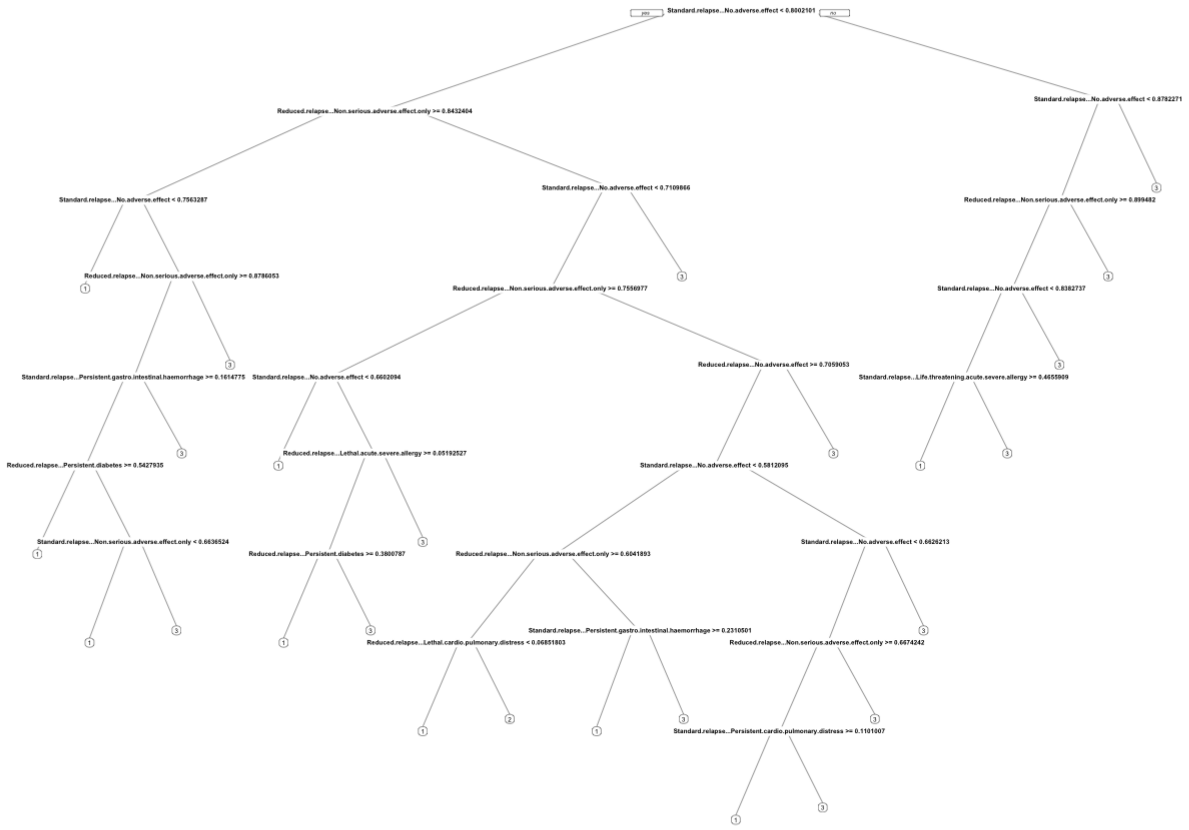
Import Processed Data

```
utilities_train <- read.csv("utilities_s1_d1_train.csv")
utilities_test <- read.csv("utilities_s1_d1_test.csv")
global_utilities <- rbind(utilities_train,utilities_test )
utilities_train$best_treatment = as.factor(utilities_train$best_treatment)
utilities_test$best_treatment = as.factor(utilities_test$best_treatment)
table(utilities_train$best_treatment)
```

```
##
##    1    2    3
## 1660  881 4459
```

Model

```
utilities_tree = rpart(best_treatment ~., data=utilities_train,cp=0.002, minbucket=10
, method="class")
prp(utilities_tree, digits = 0, varlen = 0, faclen = 0)
```


yes
Standard.relapse...No.adverse.effect < 0.8002101
no
Reduced.relapse...Non.serious.adverse.effect.only >= 0.8432404
Standard.relapse...No.adverse.effect < 0.8782271
Standard.relapse...No.adverse.effect < 0.7563287
Standard.relapse...No.adverse.effect < 0.7109866
Reduced.relapse...Non.serious.adverse.effect.only >= 0.899482
Reduced.relapse...Non.serious.adverse.effect.only >= 0.8786053
Reduced.relapse...Non.serious.adverse.effect.only >= 0.7556977
Standard.relapse...No.adverse.effect < 0.8382737
Standard.relapse...Persistent.gastro.intestinal.haemorrhage >= 0.1614775
Standard.relapse...No.adverse.effect < 0.6602094
Reduced.relapse...No.adverse.effect >= 0.7059053
Standard.relapse...Life.threatening.acute.severe.allergy >= 0.4655909
Reduced.relapse...Persistent.diabetes >= 0.5427935
Reduced.relapse...Lethal.acute.severe.allergy >= 0.05192527
Standard.relapse...No.adverse.effect < 0.5812095
Standard.relapse...Non.serious.adverse.effect.only < 0.6636524
Reduced.relapse...Persistent.diabetes >= 0.3800787
Reduced.relapse...Non.serious.adverse.effect.only >= 0.6041893
Standard.relapse...No.adverse.effect < 0.6626213
Reduced.relapse...Lethal.cardio.pulmonary.distress < 0.06851803
Standard.relapse...Persistent.gastro.intestinal.haemorrhage >= 0.2310501
Reduced.relapse...Non.serious.adverse.effect.only >= 0.6674242
Standard.relapse...Persistent.cardio.pulmonary.distress >= 0.1101007
1
2
3

DATA PERTURBATION

```
means_global = t(colMeans(global_utilities[,1:56]))
sds_global = t(colSds(as.matrix(global_utilities[,1:56])))
colnames(sds_global) <- colnames(means_global)
write.csv(means_global, "means_global.csv")
write.csv(sds_global, "sds_global.csv")
```

PERTURBATING THE TEST SET WITH 10% OF THE STANDARD DEVIATION

```
utilities_test_perturbed = data.frame(utilities_test$best_treatment)
m = length(colnames(utilities_test)) #We don't perturb the final observation
for (i in 1:(m-1)){
  perturbation = data.frame(utilities_test[,m-i] + (means_global[m-i]-rnorm(3000,mean
s_global[m-i], 0*sds_global[m-i] )))
  utilities_test_perturbed = data.frame(perturbation,utilities_test_perturbed)
  }
colnames(utilities_test_perturbed) = colnames(utilities_test)
```

```
prediction_perturbed = predict(utilities_tree, newdata = utilities_test_perturbed, ty
pe="class")
matrix_perturbed = table(utilities_test_perturbed$best_treatment, prediction_perturbe
d)
matrix_perturbed
```

```
##    prediction_perturbed
##         1    2    3
##   1  354    3  355
##   2  151    1  225
##   3  168    1 1742
```

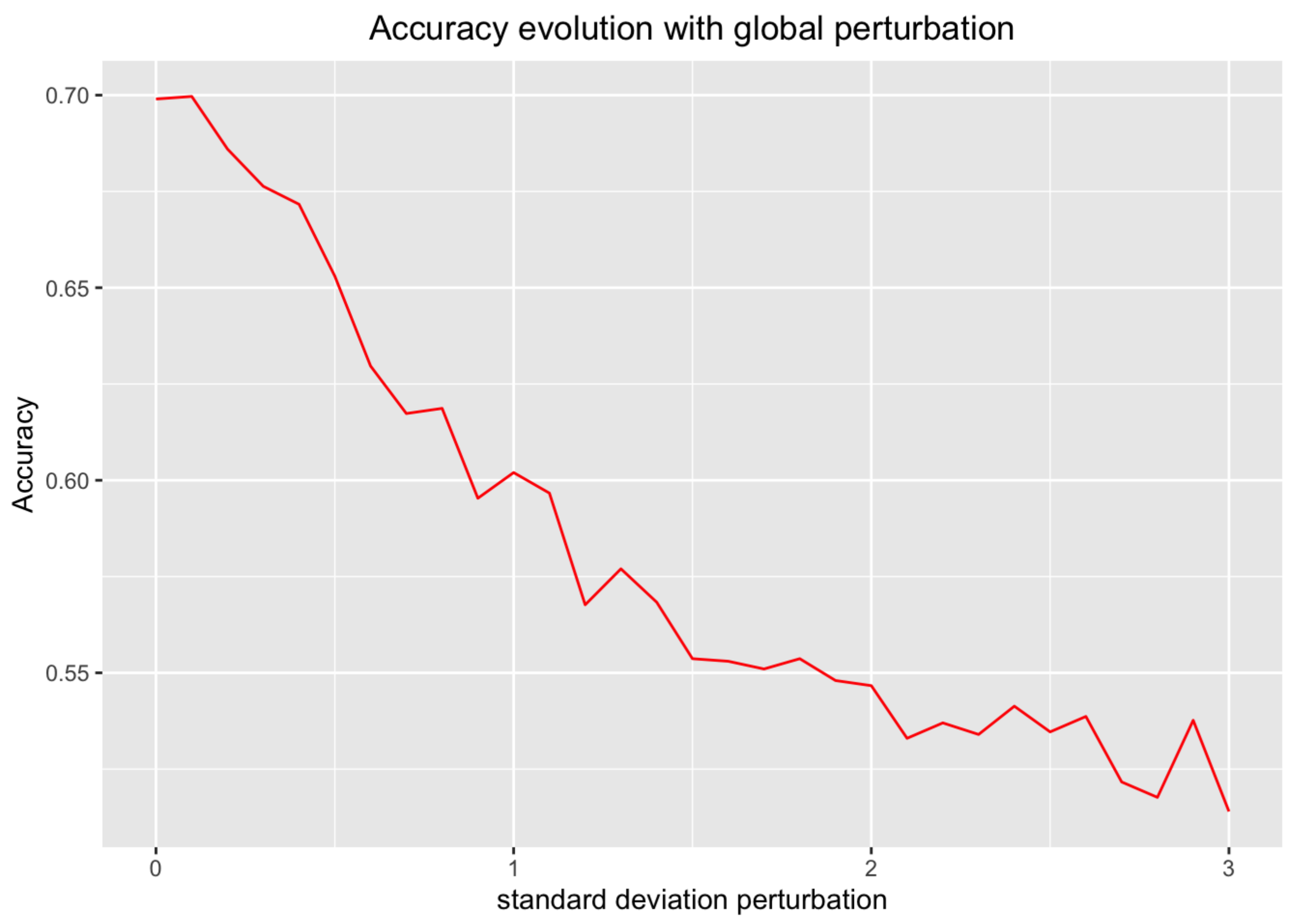
```
accuracy_pertubed = print((matrix_perturbed[1,1]+matrix_perturbed[2,2]+matrix_perturb
ed[3,3])/nrow(utilities_test))
```

```
## [1] 0.699
```

ASSESSMENT OF THE IMPACT OF A PERTURBATION ON ALL FEATURES

```
seq_pertubation = seq(from = 0, to = 3, by = 0.1)
m = length(colnames(utilities_test))
n = nrow(utilities_test) #Here it's 3000
accuracy_p <- c()
for (p in seq_pertubation){
      utilities_test_perturbed = data.frame(utilities_test$best_treatment)
      for (i in 1:(m-1)){
      perturbation = data.frame(utilities_test[,m-i] + (means_global[m-i]-rnorm(n,mea
ns_global[m-i], p*sds_global[m-i] )))
      utilities_test_perturbed = data.frame(perturbation,utilities_test_perturbed)

      }

      colnames(utilities_test_perturbed) = colnames(utilities_test)

      prediction_perturbed = predict(utilities_tree, newdata = utilities_test_perturb
ed, type="class")
      matrix_perturbed = table(utilities_test_perturbed$best_treatment, prediction_pe
rturbed)
      accuracy_pertubed = (matrix_perturbed[1,1]+matrix_perturbed[2,2]+matrix_perturb
ed[3,3])/n

      accuracy_p <- c(accuracy_p, accuracy_pertubed )

}
```

```
plot_data <- data.frame(seq_pertubation,accuracy_p)
plot_pertubation <- ggplot(plot_data) +
  geom_line(aes(x = seq_pertubation, y = accuracy_p ), color = "red")  +
  ggtitle("Accuracy evolution with global perturbation") +
  ylab("Accuracy") + xlab("standard deviation perturbation") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ggsave("accuracy_evolution_with_perturbation.png")
plot_pertubation
```



ASSESSMENT OF THE PERTUBATION WITH THE NUMBER OF LABELS PERFURBED

We select randomly a number of labels to perturb with one standard deviation, then we perturb them with a different ratio of standard deviation.

Question: How do we select these elements? We do a random forest that gives us the variable importance of each variables, then we select based on this importance.

```
rf.mod = randomForest(best_treatment ~., data=utilities_train, method="class")
importance.rf <- data.frame(imp=importance(rf.mod))
importance.rf$position <- seq(from = 1, to = 56, by = 1)
importance.rf.ordered <- importance.rf[order(-importance.rf$MeanDecreaseGini), ,drop
 = FALSE]
importance_features = data.frame(importance.rf.ordered)
write.csv(importance_features, "importance_features.csv")
```

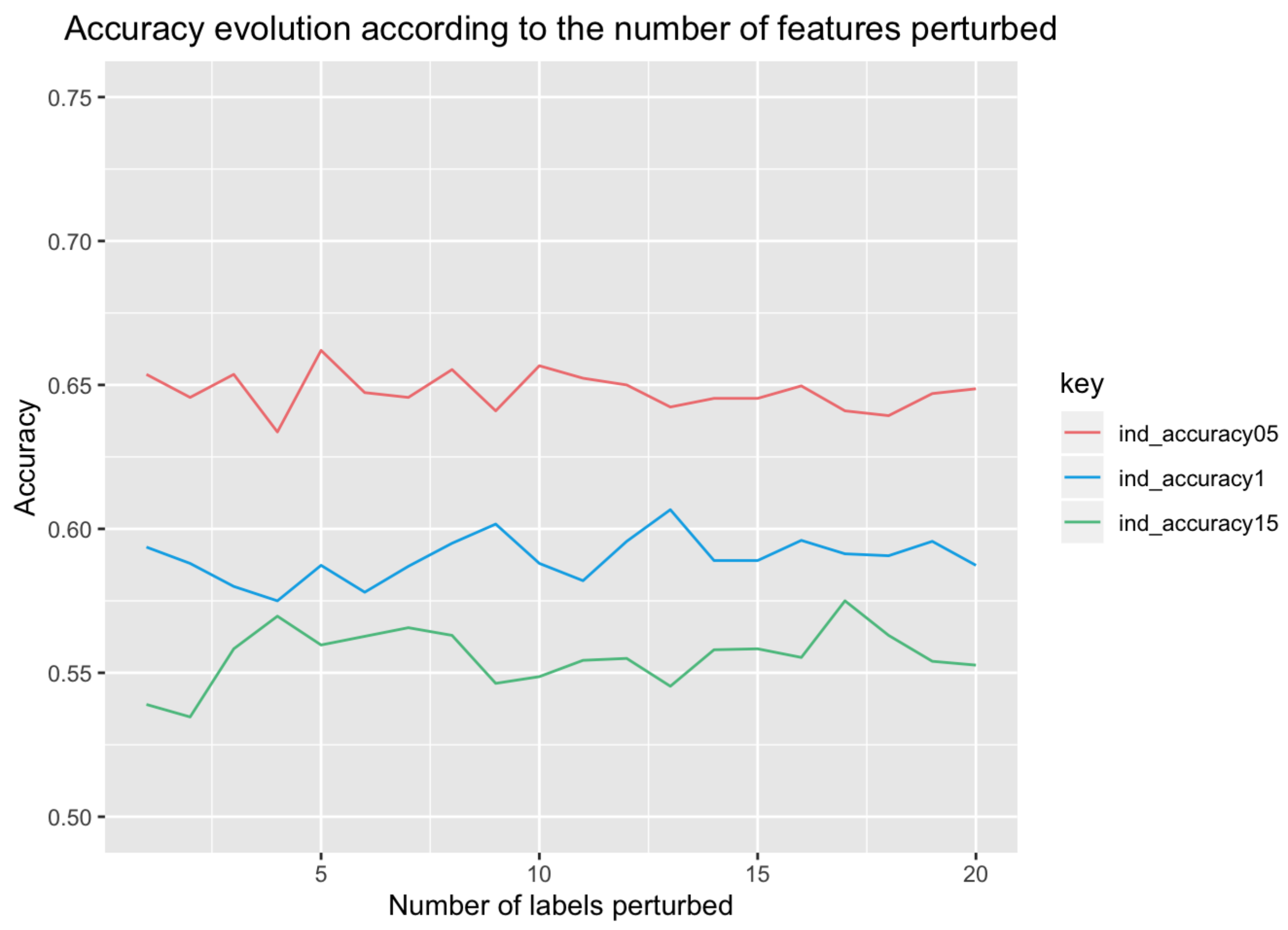
```
number_label = seq(from = 1, to = 20, by = 1)
p = 0.5 #number of sd to perturb with
ind_accuracy05 <- c()
for (number in number_label) {

    index_pertubed = importance.rf.ordered[1:number,2]

    ind_utilities_test_perturbed <- data.frame(utilities_test)
    for (index in index_pertubed){
          perturbation_ind = data.frame(utilities_test[,index] + (means_global[index]
-rnorm(n,means_global[index], p*sds_global[index])))

          ind_utilities_test_perturbed[,index] = perturbation_ind

    }

    ind_prediction_perturbed = predict(utilities_tree, newdata = ind_utilities_test_p
erturbed, type="class")

    ind_matrix_perturbed = table(ind_utilities_test_perturbed$best_treatment, ind_pre
diction_perturbed)


    ind_accuracy_pertubed = (ind_matrix_perturbed[1,1]+ind_matrix_perturbed[2,2]+ind_
matrix_perturbed[3,3])/n

    ind_accuracy05 <- c(ind_accuracy05, ind_accuracy_pertubed)
}
```

```
number_label = seq(from = 1, to = 20, by = 1)
p = 1 #number of sd to perturb with
ind_accuracy1 <- c()
for (number in number_label) {
    index_pertubed = importance.rf.ordered[1:number,2]

    ind_utilities_test_perturbed <- data.frame(utilities_test)
    for (index in index_pertubed){
          perturbation_ind = data.frame(utilities_test[,index] + (means_global[index]
-rnorm(n,means_global[index], p*sds_global[index])))

          ind_utilities_test_perturbed[,index] = perturbation_ind

    }

    ind_prediction_perturbed = predict(utilities_tree, newdata = ind_utilities_test_p
erturbed, type="class")

    ind_matrix_perturbed = table(ind_utilities_test_perturbed$best_treatment, ind_pre
diction_perturbed)


    ind_accuracy_pertubed = (ind_matrix_perturbed[1,1]+ind_matrix_perturbed[2,2]+ind_
matrix_perturbed[3,3])/n

    ind_accuracy1 <- c(ind_accuracy1, ind_accuracy_pertubed)
}
```

```r
number_label = seq(from = 1, to = 20, by = 1)
p = 1.5
ind_accuracy15 <- c()
for (number in number_label) {

    #number of sd to perturb with

    index_pertubed = importance.rf.ordered[1:number,2]

    ind_utilities_test_perturbed <- data.frame(utilities_test)
    for (index in index_pertubed){
          perturbation_ind = data.frame(utilities_test[,index] + (means_global[index]
-rnorm(n,means_global[index], p*sds_global[index])))

          ind_utilities_test_perturbed[,index] = perturbation_ind

    }

    ind_prediction_perturbed = predict(utilities_tree, newdata = ind_utilities_test_p
erturbed, type="class")

    ind_matrix_perturbed = table(ind_utilities_test_perturbed$best_treatment, ind_pre
diction_perturbed)


    ind_accuracy_pertubed = (ind_matrix_perturbed[1,1]+ind_matrix_perturbed[2,2]+ind_
matrix_perturbed[3,3])/n

    ind_accuracy15 <- c(ind_accuracy15, ind_accuracy_pertubed)
}
```

PLOT OF THE RESULTS

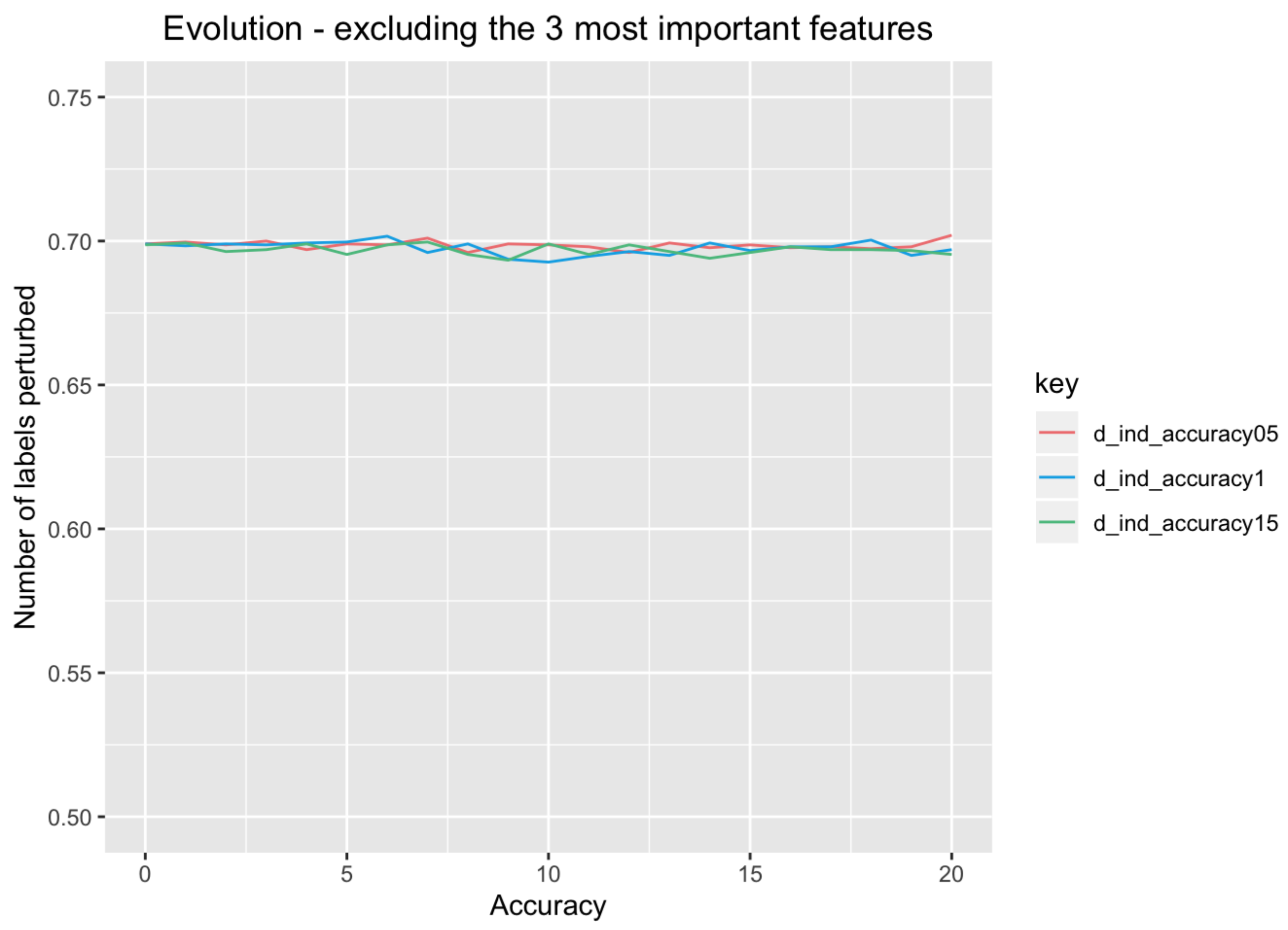
```r
data_plot_2 <- data.frame(number_label,ind_accuracy05, ind_accuracy1, ind_accuracy15)
data_plot_2 %>%
  gather(key,value, ind_accuracy05, ind_accuracy1, ind_accuracy15) %>%
   ggplot(aes(x = number_label, y=value, colour=key)) +
  geom_line() +
   ylim(0.5, 0.75) +
  ylab("Accuracy") + xlab("Number of labels perturbed") +
  ggtitle("Accuracy evolution according to the number of features perturbed") +
  theme(plot.title = element_text(hjust = 0.5)) +
   scale_color_manual(values=c("#f08080", "#00aee7", "#5ac18e") )+
  ggsave("evolution_to_features.png")
```



COMMENT: The performance doesn’t depend on the number of variables we pertub but the variables them self. Let’s perturb the variables exluding the three most important ones.

```r
number_label = seq(from = 4, to = 24, by = 1)
p = 0.5 #number of sd to perturb with
d_ind_accuracy05 <- c()
for (number in number_label) {

    index_pertubed = importance.rf.ordered[4:number,2]

    ind_utilities_test_perturbed <- data.frame(utilities_test)
    for (index in index_pertubed){
          perturbation_ind = data.frame(utilities_test[,index] + (means_global[index]
-rnorm(n,means_global[index], p*sds_global[index])))

          ind_utilities_test_perturbed[,index] = perturbation_ind

    }

    ind_prediction_perturbed = predict(utilities_tree, newdata = ind_utilities_test_p
erturbed, type="class")

    ind_matrix_perturbed = table(ind_utilities_test_perturbed$best_treatment, ind_pre
diction_perturbed)


    ind_accuracy_pertubed = (ind_matrix_perturbed[1,1]+ind_matrix_perturbed[2,2]+ind_
matrix_perturbed[3,3])/n

    d_ind_accuracy05 <- c(d_ind_accuracy05, ind_accuracy_pertubed)
}
p = 1 #number of sd to perturb with
d_ind_accuracy1 <- c()
for (number in number_label) {
    index_pertubed = importance.rf.ordered[4:number,2]

    ind_utilities_test_perturbed <- data.frame(utilities_test)
    for (index in index_pertubed){
          perturbation_ind = data.frame(utilities_test[,index] + (means_global[index]
-rnorm(n,means_global[index], p*sds_global[index])))

          ind_utilities_test_perturbed[,index] = perturbation_ind

    }

    ind_prediction_perturbed = predict(utilities_tree, newdata = ind_utilities_test_p
erturbed, type="class")

    ind_matrix_perturbed = table(ind_utilities_test_perturbed$best_treatment, ind_pre
diction_perturbed)


    ind_accuracy_pertubed = (ind_matrix_perturbed[1,1]+ind_matrix_perturbed[2,2]+ind_
matrix_perturbed[3,3])/n

    d_ind_accuracy1 <- c(d_ind_accuracy1, ind_accuracy_pertubed)
}
d_ind_accuracy15 <- c()
for (number in number_label) {
    #number of sd to perturb with

    index_pertubed = importance.rf.ordered[4:number,2]

    ind_utilities_test_perturbed <- data.frame(utilities_test)
    for (index in index_pertubed){
          perturbation_ind = data.frame(utilities_test[,index] + (means_global[index]
-rnorm(n,means_global[index], p*sds_global[index])))

          ind_utilities_test_perturbed[,index] = perturbation_ind

    }

    ind_prediction_perturbed = predict(utilities_tree, newdata = ind_utilities_test_p
erturbed, type="class")

    ind_matrix_perturbed = table(ind_utilities_test_perturbed$best_treatment, ind_pre
diction_perturbed)


    ind_accuracy_pertubed = (ind_matrix_perturbed[1,1]+ind_matrix_perturbed[2,2]+ind_
matrix_perturbed[3,3])/n

    d_ind_accuracy15 <- c(d_ind_accuracy15, ind_accuracy_pertubed)
}
```

```
number_label_3 <- number_label-4
data_plot_3 <- data.frame(number_label_3,d_ind_accuracy05, d_ind_accuracy1, d_ind_acc
uracy15)
data_plot_3 %>%
  gather(key,value, d_ind_accuracy05, d_ind_accuracy1, d_ind_accuracy15) %>%
   ggplot(aes(x = number_label_3, y=value, colour=key)) +
  geom_line() +
   ylim(0.5, 0.75) +
  xlab("Accuracy") + ylab("Number of labels perturbed") +
  ggtitle("Evolution - excluding the 3 most important features") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ggsave("evolution_to_features_2.png") +
  scale_color_manual(values=c("#f08080", "#00aee7", "#5ac18e"))
```



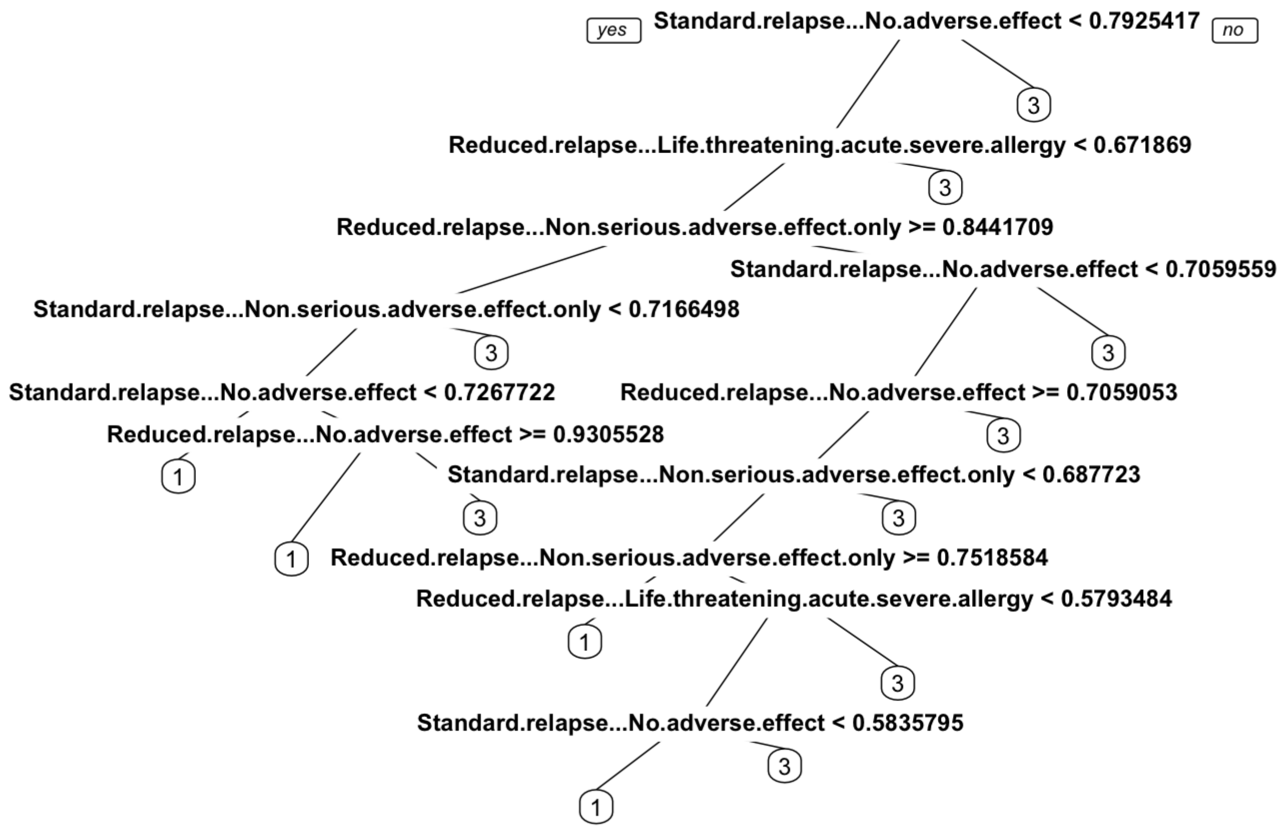
COMMENT: This has very important implications, it implies that we should control only the three first variables to be robust to data perturbation. This three variables are : Standard.relapse…No.adverse.effect, Standard.relapse…Non.serious.adverse.effect.only, Reduced.relapse…Non.serious.adverse.effect.only

MODEL’S ROBUSTIFICATION

We robustify by generating pertubation in the data set for this three variables.

```
means_train = t(colMeans(global_utilities[,1:56]))
sds_train = t(colSds(as.matrix(global_utilities[,1:56])))
colnames(sds_train) <- colnames(means_global)
index_important = importance.rf.ordered[1:3,2]
utilities_train_05 <- data.frame(utilities_train)
utilities_train_1 <- data.frame(utilities_train)
utilities_train_15 <- data.frame(utilities_train)
n = nrow(utilities_train)
for (index in index_important){

      perturbation_ind_05 = data.frame(utilities_train[,index] + (means_train[index]-
rnorm(n,means_train[index], 0.5*sds_train[index])))
      perturbation_ind_1 = data.frame(utilities_train[,index] + (means_train[index]-r
norm(n,means_train[index], 1*sds_train[index])))
      perturbation_ind_15 = data.frame(utilities_train[,index] + (means_train[index]-
rnorm(n,means_train[index], 1.5*sds_train[index])))

      utilities_train_05[,index] = perturbation_ind_05
      utilities_train_1[,index] = perturbation_ind_1
      utilities_train_15[,index] = perturbation_ind_15
}
robust_utilities_train = rbind(utilities_train, utilities_train_05, utilities_train_
1, utilities_train_15 )
write.csv(robust_utilities_train, "robust_utilities_train.csv")
```

```
robust_utilities_tree = rpart(best_treatment ~., data=robust_utilities_train,cp=0.002
, minbucket=10, method="class")
prp(robust_utilities_tree, digits = 0, varlen = 0, faclen = 0)
```


yes
Standard.relapse...No.adverse.effect < 0.7925417
no
3
Reduced.relapse...Life.threatening.acute.severe.allergy < 0.671869
3
Reduced.relapse...Non.serious.adverse.effect.only >= 0.8441709
Standard.relapse...No.adverse.effect < 0.7059559
Standard.relapse...Non.serious.adverse.effect.only < 0.7166498
3
3
Standard.relapse...No.adverse.effect < 0.7267722
Reduced.relapse...No.adverse.effect >= 0.7059053
Reduced.relapse...No.adverse.effect >= 0.9305528
3
1
Standard.relapse...Non.serious.adverse.effect.only < 0.687723
3
3
1
Reduced.relapse...Non.serious.adverse.effect.only >= 0.7518584
Reduced.relapse...Life.threatening.acute.severe.allergy < 0.5793484
1
3
Standard.relapse...No.adverse.effect < 0.5835795
3
1

FIRST ASSESSMENT OF THE ROBUSTIFIED MODEL

```
prediction = predict(robust_utilities_tree, newdata = utilities_test, type="class")
matrix = table(utilities_test$best_treatment, prediction)
matrix
```

```
##    prediction
##        1    2    3
##   1  243    0  469
##   2   77    0  300
##   3   75    0 1836
```

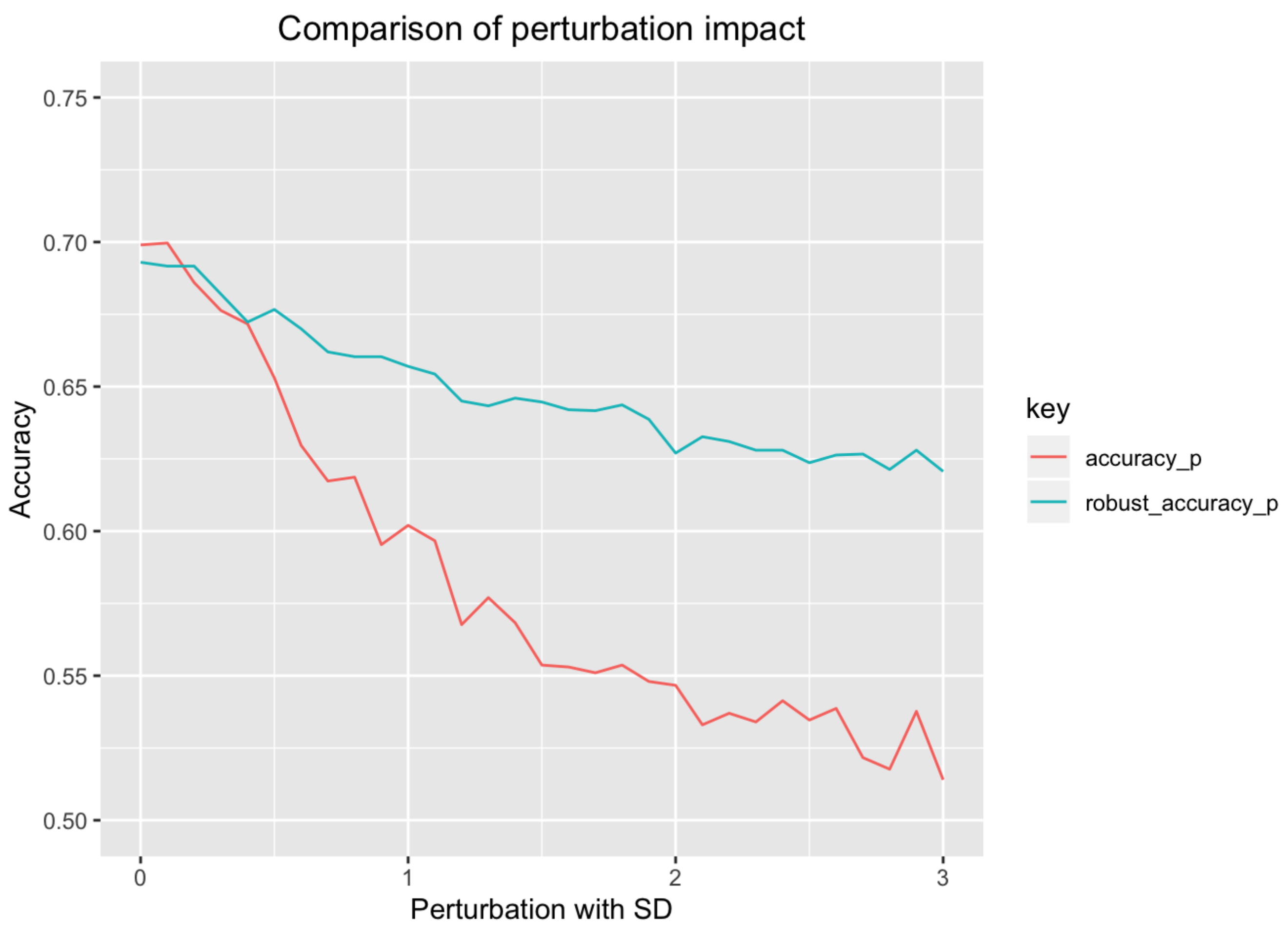
```
print((matrix[1,1]+matrix[2,2]+matrix[3,3])/nrow(utilities_test))
```

```
## [1] 0.693
```

ASSESSMENT OF THE IMPACT OF A PERTURBATION ON ALL FEATURES WITH ROBUST DATA

```
seq_pertubation = seq(from = 0, to = 3, by = 0.1)
m = length(colnames(utilities_test))
n = nrow(utilities_test) #Here it's 3000
robust_accuracy_p <- c()
for (p in seq_pertubation){
      utilities_test_perturbed = data.frame(utilities_test$best_treatment)
      for (i in 1:(m-1)){
      perturbation = data.frame(utilities_test[,m-i] + (means_global[m-i]-rnorm(n,mea
ns_global[m-i], p*sds_global[m-i] )))
      utilities_test_perturbed = data.frame(perturbation,utilities_test_perturbed)

      }

      colnames(utilities_test_perturbed) = colnames(utilities_test)

      prediction_perturbed = predict(robust_utilities_tree, newdata = utilities_test_
perturbed, type="class")
      matrix_perturbed = table(utilities_test_perturbed$best_treatment, prediction_pe
rturbed)
      accuracy_pertubed = (matrix_perturbed[1,1]+matrix_perturbed[2,2]+matrix_perturb
ed[3,3])/n

      robust_accuracy_p <- c(robust_accuracy_p, accuracy_pertubed )

}
```

```
plot_data <- data.frame(seq_pertubation,robust_accuracy_p, accuracy_p)
plot_data %>%
  gather(key,value, robust_accuracy_p, accuracy_p) %>%
   ggplot(aes(x = seq_pertubation, y=value, colour=key)) +
  geom_line() +
   ylim(0.5, 0.75) +
  xlab("Perturbation with SD") + ylab("Accuracy") +
  ggtitle("Comparison of perturbation impact") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ggsave("robust_accuracy_evolution_with_perturbation.png")
```



COMMENT: Robustifying the data with only the three most important features gives impressive resistance to data perturbation. The first lower performance is due to a more complex data generated while robustifying the CART, but where the data is perturbed at higher values, the robust model cleary outperforms the basic one.

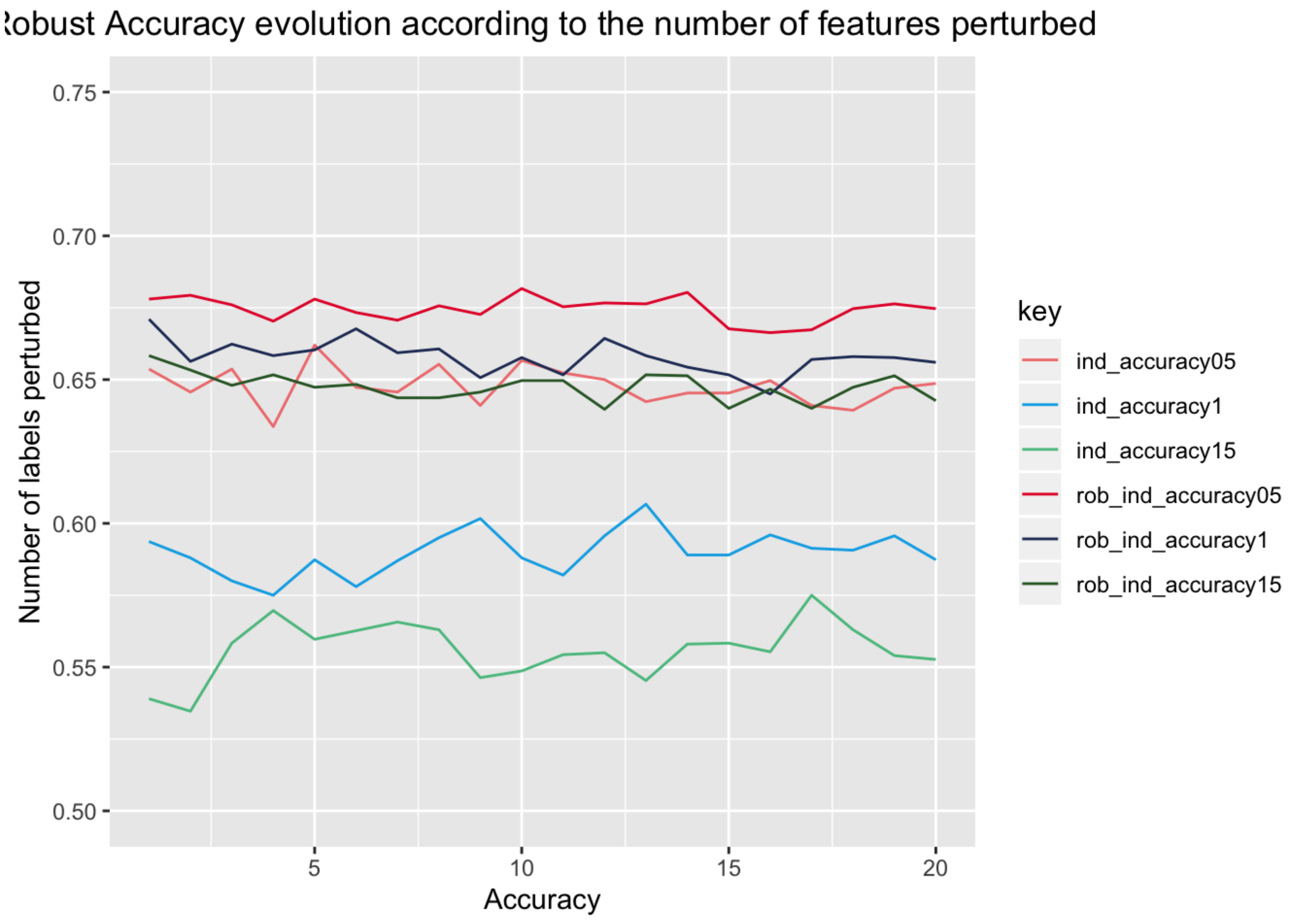
```r
number_label = seq(from = 1, to = 20, by = 1)
p = 0.5 #number of sd to perturb with
rob_ind_accuracy05 <- c()
for (number in number_label) {

    index_pertubed = importance.rf.ordered[1:number,2]

    ind_utilities_test_perturbed <- data.frame(utilities_test)
    for (index in index_pertubed){
          perturbation_ind = data.frame(utilities_test[,index] + (means_global[index]
-rnorm(n,means_global[index], p*sds_global[index])))

          ind_utilities_test_perturbed[,index] = perturbation_ind

    }

    ind_prediction_perturbed = predict(robust_utilities_tree, newdata = ind_utilities
_test_perturbed, type="class")

    ind_matrix_perturbed = table(ind_utilities_test_perturbed$best_treatment, ind_pre
diction_perturbed)


    ind_accuracy_pertubed = (ind_matrix_perturbed[1,1]+ind_matrix_perturbed[2,2]+ind_
matrix_perturbed[3,3])/n

    rob_ind_accuracy05 <- c(rob_ind_accuracy05, ind_accuracy_pertubed)
}
```

```r
number_label = seq(from = 1, to = 20, by = 1)
p = 1 #number of sd to perturb with
rob_ind_accuracy1 <- c()
for (number in number_label) {
    index_pertubed = importance.rf.ordered[1:number,2]

    ind_utilities_test_perturbed <- data.frame(utilities_test)
    for (index in index_pertubed){
          perturbation_ind = data.frame(utilities_test[,index] + (means_global[index]
-rnorm(n,means_global[index], p*sds_global[index])))

          ind_utilities_test_perturbed[,index] = perturbation_ind

    }

    ind_prediction_perturbed = predict(robust_utilities_tree, newdata = ind_utilities
_test_perturbed, type="class")

    ind_matrix_perturbed = table(ind_utilities_test_perturbed$best_treatment, ind_pre
diction_perturbed)


    ind_accuracy_pertubed = (ind_matrix_perturbed[1,1]+ind_matrix_perturbed[2,2]+ind_
matrix_perturbed[3,3])/n

    rob_ind_accuracy1 <- c(rob_ind_accuracy1, ind_accuracy_pertubed)
}
```

```r
number_label = seq(from = 1, to = 20, by = 1)
p = 1.5
rob_ind_accuracy15 <- c()
for (number in number_label) {

    #number of sd to perturb with

    index_pertubed = importance.rf.ordered[1:number,2]

    ind_utilities_test_perturbed <- data.frame(utilities_test)
    for (index in index_pertubed){
          perturbation_ind = data.frame(utilities_test[,index] + (means_global[index]
-rnorm(n,means_global[index], p*sds_global[index])))

          ind_utilities_test_perturbed[,index] = perturbation_ind

    }

    ind_prediction_perturbed = predict(robust_utilities_tree, newdata = ind_utilities
_test_perturbed, type="class")

    ind_matrix_perturbed = table(ind_utilities_test_perturbed$best_treatment, ind_pre
diction_perturbed)


    ind_accuracy_pertubed = (ind_matrix_perturbed[1,1]+ind_matrix_perturbed[2,2]+ind_
matrix_perturbed[3,3])/n

    rob_ind_accuracy15 <- c(rob_ind_accuracy15, ind_accuracy_pertubed)
}
```

PLOT OF THE RESULTS

```r
data_plot_5 <- data.frame(number_label,ind_accuracy05, ind_accuracy1, ind_accuracy15,
rob_ind_accuracy05, rob_ind_accuracy1, rob_ind_accuracy15)
data_plot_5 %>%
  gather(key,value, ind_accuracy05, ind_accuracy1, ind_accuracy15,rob_ind_accuracy05,
rob_ind_accuracy1, rob_ind_accuracy15) %>%
   ggplot(aes(x = number_label, y=value, colour=key)) +
  geom_line() +
   ylim(0.5, 0.75) +
  xlab("Accuracy") + ylab("Number of labels perturbed") +
  ggtitle("Robust Accuracy evolution according to the number of features perturbed")
 +
  theme(plot.title = element_text(hjust = 0.5)) +
  ggsave("robust_evolution_to_features.png") +
  scale_color_manual(values=c("#f08080", "#00aee7", "#5ac18e","#e51635", "#273b66",
"#336633"))
```



Comparison with Multinomial glmnet

```r
multi_threshold<- function(pred, actual_values){
  sequence = seq(0,1, by=0.05)
  n = length(actual_values)
  values = c()
  accuracy_max = 0
  for (i in 1:length(sequence)){
      prediction_val = c()
      thresh = sequence[i]
      for (j in 1:nrow(pred)) {
          val1 = pred[j,1]
          val2 = pred[j,2]
          val3 = pred[j,3]


          if(thresh>val2){
            prediction_val <- c(prediction_val, 3)
          }
          if(thresh<=val2){
            if (thresh<=val1){
              prediction_val <- c(prediction_val, 1)
            }
            else{
              prediction_val <- c(prediction_val, 2)
            }
          }


      }

    table_val = table(actual_values, prediction_val)
    values_pred = as.numeric(colnames(table_val))
    for (k in (1:length(values_pred))){
      good_pred = table_val[values_pred[k],k]
    }

     accuracy_thresh = good_pred/n

    if (accuracy_thresh>accuracy_max){
       best_thresh = thresh
       accuracy_max = accuracy_thresh
      best_pred = prediction_val
     }

  }
   return( best_thresh )
}
threshold_accuracy <- function(pred, actual_values, thresh){

  n = length(actual_values)
      prediction_val = c()
      for (j in 1:nrow(pred)) {
          val1 = pred[j,1]
          val2 = pred[j,2]
          val3 = pred[j,3]


          if(thresh>val2){
            prediction_val <- c(prediction_val, 3)
          }
          if(thresh<=val2){
            if (thresh<=val1){
              prediction_val <- c(prediction_val, 1)
            }
            else{
              prediction_val <- c(prediction_val, 2)
            }
          }



      }

    table_val = table(actual_values, prediction_val)
    values_pred = as.numeric(colnames(table_val))
    for (k in (1:length(values_pred))){
      good_pred = table_val[values_pred[k],k]
    }

     accuracy_thresh = good_pred/n

   return( accuracy_thresh )
}
threshold_prediction <- function(pred, actual_values, thresh){
      n = length(actual_values)
      prediction_val = c()
      for (j in 1:nrow(pred)) {
          val1 = pred[j,1]
          val2 = pred[j,2]
          val3 = pred[j,3]
          if(thresh>val2){
            prediction_val <- c(prediction_val, 3)
          }
          if(thresh<=val2){
            if (thresh<=val1){
              prediction_val <- c(prediction_val, 1)
            }
            else{
              prediction_val <- c(prediction_val, 2)
            }
          }
      }
   return( prediction_val )
}
```

```r
x.train = model.matrix(best_treatment ~., data=utilities_train)
x.test = model.matrix(best_treatment ~., data=utilities_test)
y.train = utilities_train$best_treatment
y.test = utilities_test$best_treatment
set.seed(1)
#glm_treatment.cv <- cv.glmnet(x.train, y.train, alpha=1, family="multinomial", inter
cept = F, nfolds = 5)

#glm_treatment.min <- glm_treatment.cv$lambda.min
#glm_treatment.1se <- glm_treatment.cv$lambda.1se

glm_treatment.min = 0.001029396
glm_treatment.1se = 0.01156345
glm_treatment <- glmnet(x.train,y.train,alpha=1,family="multinomial", lambda = glm_tr
eatment.min, intercept = F)
# Definition of threshold
pred_in <- predict(glm_treatment,newx=x.train,type = 'response')
pred_in.df <- data.frame(pred_in)
best_threshold <- multi_threshold(pred_in.df, y.train)
### OUT-SAMPLE
pred <- data.frame(predict(glm_treatment,newx=x.test,type = 'response'))
best_out_accuracy <- threshold_accuracy(pred, y.test, 0.5)
```

```r
multi_pred <- function(pred){
    prediction <- c()
    for (i in 1:nrow(pred)) {
      val <- c(pred[i,1], pred[i,2], pred[i,3])
      prediction <- c(prediction, which.max(val))
    }
    return(prediction)
}
```

```
seq_pertubation = seq(from = 0, to = 3, by = 0.1)
m = length(colnames(utilities_test))
n = nrow(utilities_test) #Here it's 3000
robust_accuracy_p <- c()
for (p in seq_pertubation){
      utilities_test_perturbed = data.frame(utilities_test$best_treatment)
      for (i in 1:(m-1)){
      perturbation = data.frame(utilities_test[,m-i] + (means_global[m-i]-rnorm(n,mea
ns_global[m-i], p*sds_global[m-i] )))
      utilities_test_perturbed = data.frame(perturbation,utilities_test_perturbed)

      }

      colnames(utilities_test_perturbed) = colnames(utilities_test)

      x.test.perturbed = model.matrix(best_treatment ~., data=utilities_test_perturbe
d)
      y.test.perturbed  = utilities_test_perturbed$best_treatment

      prediction_perturbed = data.frame(predict(glm_treatment,newx=x.test.perturbed,t
ype = 'response'))

      prediction_val = multi_pred(prediction_perturbed)

      table_val = table(y.test.perturbed, prediction_val)
      values_pred = as.numeric(colnames(table_val))
      for (k in (1:length(values_pred))){
            good_pred = table_val[values_pred[k],k]
       }

      accuracy_thresh_perturbed = good_pred/n

      robust_accuracy_p <- c(robust_accuracy_p, accuracy_thresh_perturbed )

}
write.csv(robust_accuracy_p, "robust_accuracy_p.csv")
write.csv(seq_pertubation, "seq_pertubation.csv")
```

NON REGULARIZED LOGISTIC REGRESSION PERFORMANCE

```r
x.train = model.matrix(best_treatment ~., data=utilities_train)
x.test = model.matrix(best_treatment ~., data=utilities_test)
y.train = utilities_train$best_treatment
y.test = utilities_test$best_treatment
set.seed(1)
#glm_treatment.cv <- cv.glmnet(x.train, y.train, alpha=1, family="multinomial", inter
cept = F, nfolds = 5)

#glm_treatment.min <- glm_treatment.cv$lambda.min
#glm_treatment.1se <- glm_treatment.cv$lambda.1se

glm_treatment.min = 0.0
glm_treatment <- glmnet(x.train,y.train,alpha=1,family="multinomial", lambda = glm_tr
eatment.min, intercept = F)
# Definition of threshold
pred_in <- predict(glm_treatment,newx=x.train,type = 'response')
pred_in.df <- data.frame(pred_in)
best_threshold <- multi_threshold(pred_in.df, y.train)
### OUT-SAMPLE
pred <- data.frame(predict(glm_treatment,newx=x.test,type = 'response'))
best_out_accuracy <- threshold_accuracy(pred, y.test, 0.5)
```

```r
multi_pred <- function(pred){
    prediction <- c()
    for (i in 1:nrow(pred)) {
      val <- c(pred[i,1], pred[i,2], pred[i,3])
      prediction <- c(prediction, which.max(val))
    }
    return(prediction)
}
```

```r
seq_pertubation = seq(from = 0, to = 3, by = 0.1)
m = length(colnames(utilities_test))
n = nrow(utilities_test) #Here it's 3000
robust_accuracy_p_nr <- c()
for (p in seq_pertubation){
      utilities_test_perturbed = data.frame(utilities_test$best_treatment)
      for (i in 1:(m-1)){
      perturbation = data.frame(utilities_test[,m-i] + (means_global[m-i]-rnorm(n,mea
ns_global[m-i], p*sds_global[m-i] )))
      utilities_test_perturbed = data.frame(perturbation,utilities_test_perturbed)

      }

      colnames(utilities_test_perturbed) = colnames(utilities_test)

      x.test.perturbed = model.matrix(best_treatment ~., data=utilities_test_perturbe
d)
      y.test.perturbed  = utilities_test_perturbed$best_treatment

      prediction_perturbed = data.frame(predict(glm_treatment,newx=x.test.perturbed,t
ype = 'response'))

      prediction_val = multi_pred(prediction_perturbed)

      table_val = table(y.test.perturbed, prediction_val)
      values_pred = as.numeric(colnames(table_val))
      for (k in (1:length(values_pred))){
            good_pred = table_val[values_pred[k],k]
       }

      accuracy_thresh_perturbed = good_pred/n

      robust_accuracy_p_nr <- c(robust_accuracy_p_nr, accuracy_thresh_perturbed )

}
write.csv(robust_accuracy_p_nr, "robust_accuracy_p_nr.csv")
write.csv(seq_pertubation, "seq_pertubation_nr.csv")
```