STAT 616- PROJECT REPORT
Elif DOĞAN DAR- 2206753
Summer, 2021

DATA DESCRIPTION AND PRE-PROCESSING

Nhanes data set contains 65 variables and 3831 samples. In numerical variables there are approximately 2.2 % missing values. Although it seems like categorical variables don't have missing values, they are entries which are encoded as "Unknown". I checked the variable meanings to see if they actually mean "others". But there were already categories as "Others", for instance in COUNTRY variable. Therefore, I was convinced that these unknown entries are actually missing. A confirmation can be requested from the data providers. After this replacement, actual percentage of the missing values is 7.9 %.

Categorical variables were kept as character strings, I transformed them into factors. From the data dictionary, I get the information of relations between some variables. They are as follows:

- BMI- MET4: (MET4=Yes if BMI >=25) and (No otherwise).
- WAIST- MET1A: (MET1E=Yes if (WAIST>= 102 and GENDER=Men
  WAIST>=89 and GENDER= Women))
  MET1E= No otherwise.
- MET2-FAST: (MET2=Yes if FAST>=0.3) and (MET2=No otherwise).
- BMIGRP is the categorized version of BMI variable.
- AGEGRP is the categorized version of RIDAGEYR variable.
- SEQN doesn't have any prediction value, just an identification number for the samples.
- POVERTY is derived from the variable HHINCOME.
- It seemed like MET5 takes a certain value when ALT, AST and ALP values are in a certain range, but when I plotted them, I couldn't confirm this. There were intersections, more information can be requested from the data providers.
- There are two variables DEPRESS and DEPRESS_SCORE. Better to check if they cause multicollinearity. But if they are scores from separate tests or evaluations, they might not agree on every cases and they might show different aspects of depression, caution is necessary. Since I don't know the details and there doesn't seem to be a relation shown in the dictionary, I keep them both.

Since it causes redundancy in the data, categorized versions of the variables are dropped from the study. They are used for exploratory data analysis though, since it is easier to notice relations among variables through categorized versions.

Clusters: When we look at the correlation among numerical variables we see some clusters. AST, ALT and FAST variables all have 0.7 or higher correlation among them. AGE, NFS and FIB4 all have correlations around 0.7. Lastly, GLUCOSE, HBA1C and IFG has 0.7 or higher correlation among them.

However, we should keep in mind that high Pearson correlation coefficient is necessary but not sufficient to show linear dependency among the variables. Scatterplots should be used for additional visual inspection before taking any action.

WAIST and BMI variables have a correlation coefficient of 0.9. Scatterplot also supports the positive strong linear relationship among them. It would be safe to drop WAIST variable to avoid multicollinearity for the models which are not immune. Also, there might be strong non-linear relations among the variables which cannot be captured by Pearson correlation. We are hoping the models to capture these for us.

An inspection on GLUCOSE and IFG relationship reveals that, all entries with 106 and higher GLUCOSE values has IFG value 0 and all entries with GLUCOSE value 126 and higher gives IFG

value 1. Most probably, IFG is categorized version of the GLUCOSE variable. More information can be requested from the data providers.

EXPLORATORY DATA ANALYSIS

Since data set is pretty large for the exploratory data analysis, I automated graph building process. To examine categorical variables I used geom.bar() function. First, simple bar graph, then by faceting according to NAFLD. Finally, by using propensities instead of number of cases.
To see relations between numeric variables, I applied ggplot's geom_point() function with coloring with NAFLD variable. To see relations between numerical and categorical variables, I applied geom_boxplot() function, again with color NAFLD. You can find some representative graphs in the appendix.

Through exploratory data analysis, I noticed that some variables has near zero information. Namely, AUTO_HEP, VIR_HEP, LIV_CIR, LIV_FIB, WGT_SURG, TYPE2DB . For example, AUTO_HEP has only 2 Yes entries compared to 3819 No and 10 unknowns. There are two other variables CHF(88 Yes, 3722 No) and LIV_FAT(92 Yes, 3730 No) too. They are unbalanced but since they might be containing crucial information, I kept them. Further investigation can be done with the help of domain experts.

In the propensity plots I have also seen that some variables have NA values which always belong to non-nafld case. But a closer examination revealed that they are too less to conclude that missingness are not random. For example, in HTNRT variable all NA values belongs to non-nafld class but there are only 6 NA values in total. Caution is needed while working with propensities.

MISSING DATA DISTRIBUTION AND IMPUTATION

Approximately 8% of the data is missing and distribution of missingness to the variables is not uniform. Variables with more than 15% of missingness are as follows: LIVER_STILL (96%), TYPE2DB(78%), DBPREV(74%) , GLUCOSE(53%) , MET1A (45%). These variables are excluded from the data since it is not safe to impute variables with that high missingness. GLUCOSE has approximately 0.9 correlation with HBA1C, therefore information coming from it will already be retained because of this relationship. Similarly, MET1A is the categorized version of the interaction of WAIST and GENDER variables, therefore, that information is also somewhat retained.

There are 191 samples with more than 25% of missing data. Since our data set is large enough, we can exclude these samples from the prediction task. It would have been wiser to prepare a dissimilarity plot showing similarity of these 191 samples and others. If they are similar , then we can conclude that it is safer to exclude them. The reason behind is that they wouldn't represent a particular subset of the data which has to be represented even in the expense of some wrong imputations.

To see the missing data patterns, I applied md.pattern() from mice package. I have seen that in 387 cases only WGTSURG is missing. In 319 cases POVERTY and HHINCOME is missing, which makes sense since POVERTY is derived from HHINCOME. In 110 cases DEPRESS and DEPRESS_SCORE are missing which also makes sense. In 46 cases MET1D, SISTOLIC and DISTOLIC are missing, there might be a medical reason behind it, these variables might be related. And in many other cases 46+31+23+25+.. there are blocks where many medical lab values are missing all together as blocks. Overall, there doesn't seem to be a reason to think that imputation is not doable.

I also checked md.pairs() function which gives missingness patterns of pairs. Rr matrix shows the cases where both variables are observed, mm matrix shows where both variables are missing and mr,rm matrices show the number of cases which one of them is observed and one of them is missing. Just for a demonstration, I share cases for FAST and MET2 pair (rr:3624, rm:0, mr:0, mm:207) . Since MET2 is derived from FAST, it makes sense that they are either or observed together. In the case of

HHINCOME and POVERTY pair(rr:3284, rm:2, mr:44, mm:547). Since POVERTY is derived from HHINCOME and another hidden variable, it is expected that 44 cases are missing where Poverty is missing although HHINCOME exists; it means that hidden variable is missing. However, it is actually impossible to be able to derive POVERTY when HHINCOME is missing, does it mean that there are some random deletings at the data or a technical problem? Since there are only 2 cases like that, it doesn't affect our prediction problem tough.

I used "mice" package for the imputations. First, I checked the predictor matrix and noticed that COHORT variable is not used as an imputation predictor. One possible reason is the collinearity, when I checked, I saw that COHORT variable has exactly the same info as the dependent variable. So, it is meaningless to build a model when we know all the information already, therefore I excluded it from the data. Also, I manually excluded SEQN variable from the imputation prediction by setting its column 0 in the predictor matrix. I made 5 imputations with maximum 10 iterations. I checked the density plots and the stripplots to see if imputations are meaningful or not. There doesn't seem to be any problem with densities. But there were some convergence problems. By dropping SEQN, COHORT and other uninformative variables I solved most of the convergence problems. Number of iterations can be increased for better results.

Although densities and stripplots seemed fine, in the DEPRESS and DEPRESS_SCORE variables, we see that the imputations make the means higher. This might mean that most of the missing depression scores are from the people with higher depression. Can it be possible that people who have higher depression values tend to answer less? This insight can be shared with the data providers.

RANDOM FOREST IMPLEMENTATION

First I divided data randomly to training and test sets, 2/3 and 1/3 respectively. I used caret package to apply 10 fold cross validation. According to accuracy rises fast from mtry=2 (accuracy=0.78) untill mtry=31(accuracy=0.85) and then rises slowly till mtry=60(accuracy=0.86), which was automatically chosen by the model. However, to get rid of possible overfitting, I chose mtry=31 in the final model. Test accuracy was 0.88 for this model.

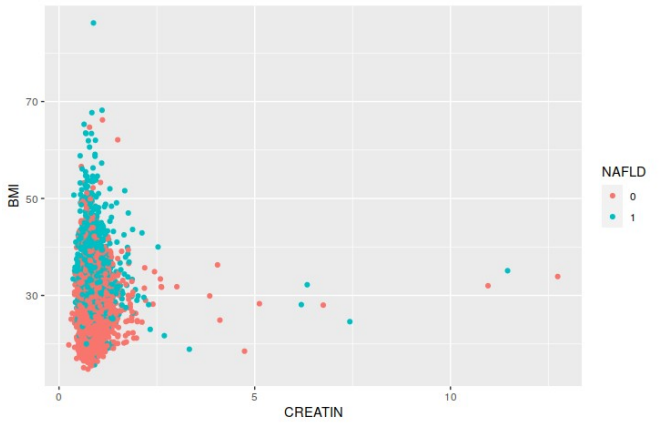
After that, I did 10 fold cross validation to check number of trees for values 50,100,150,200,250,300,350, 400, 450 and 500. After 50 it seems like stabilized but for generalization purposes, I used 200 in the final model. With this model test accuracy is 0.88 and AUC value is 0.93.

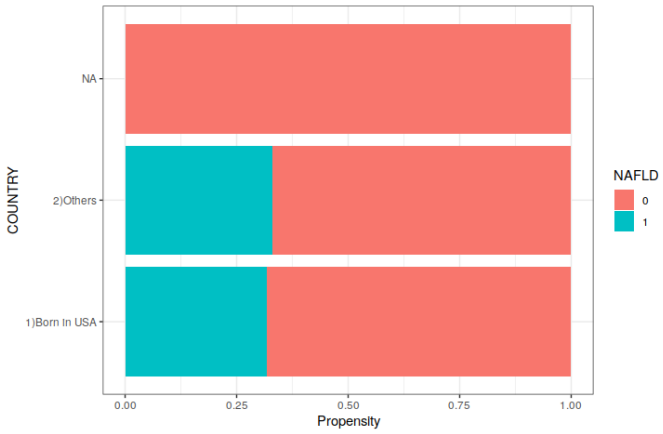
I checked variable importanca values an saw that FAST, BMI, AST, HBA1C and TRIGLY are the most important variables.

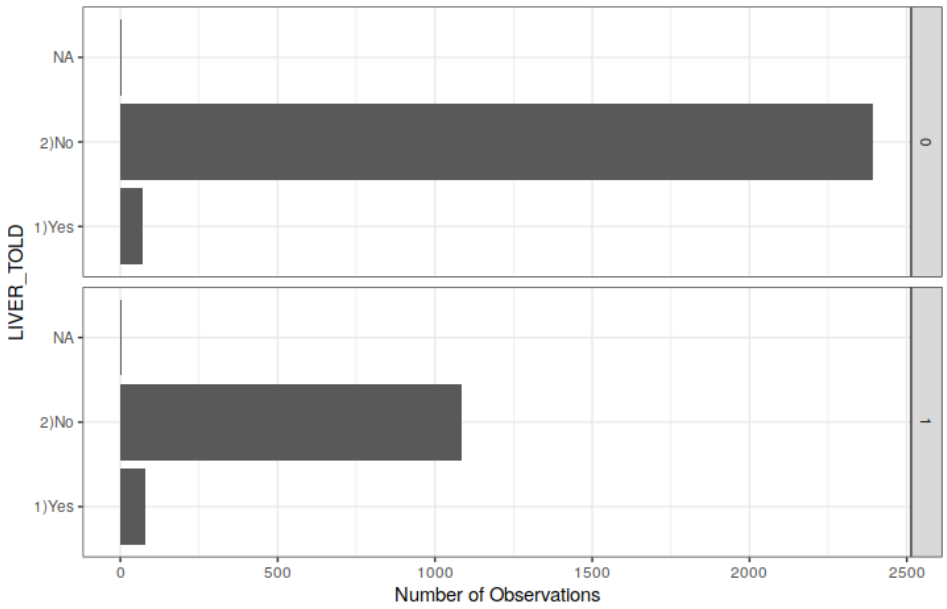
Also, partial dependence plot of FAST shows that probability of having NAFLD gets higher as FAST increases and becomes extremely high after FAST value reaches 0.35. Partial dependence plot of BMI shows similar trend, getting extremely high after BMI 40, and almost 1 after BMI 50. Plots can be found in the appendix.

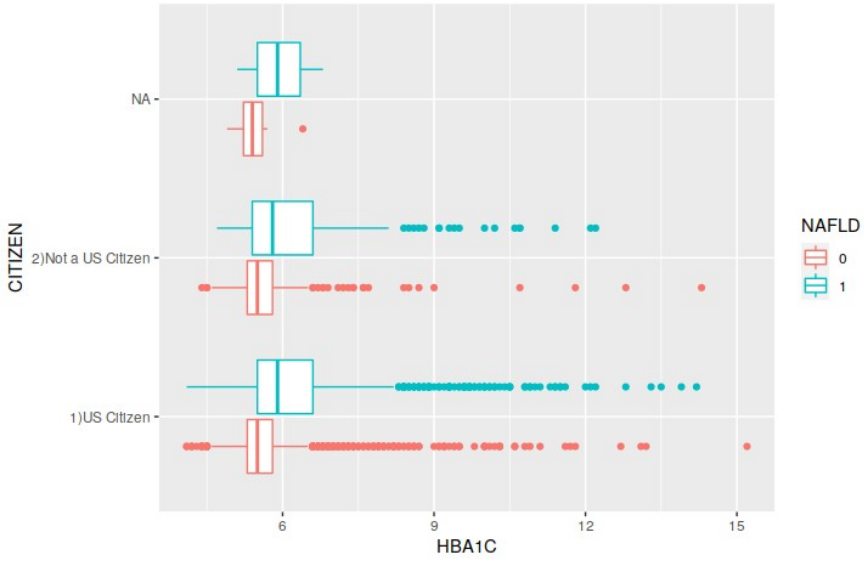
I finally checked MDSplot and saw that classes are well separated with some intersection. Plot can be found in appendix.

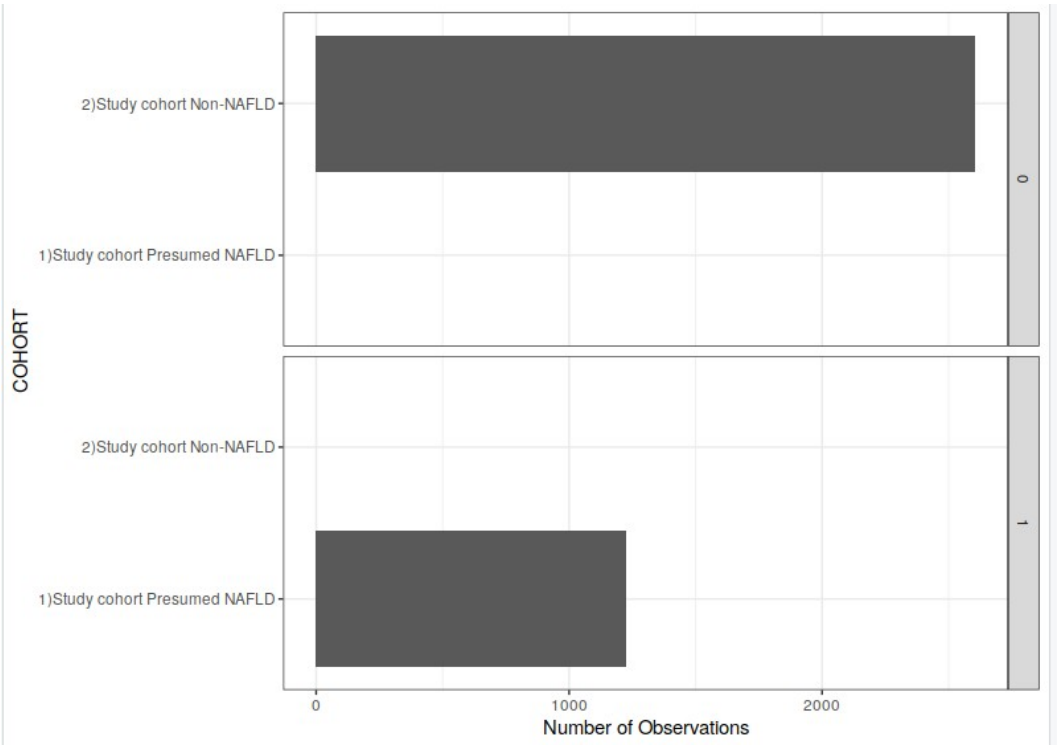
APPENDIX

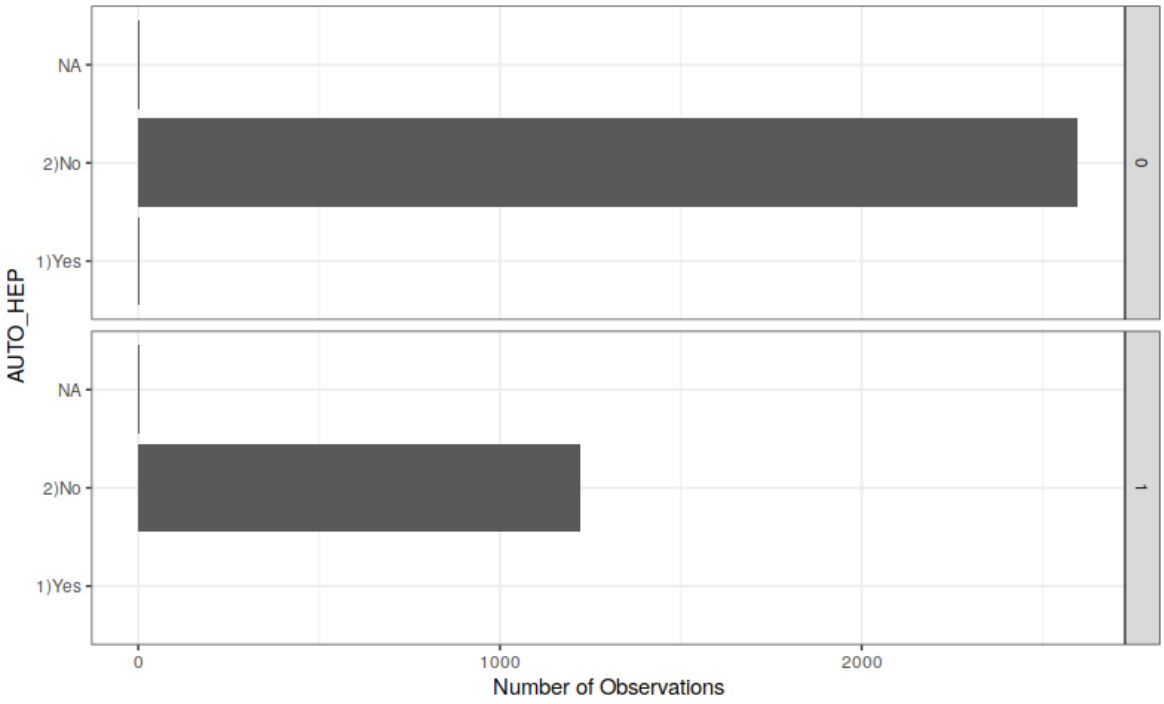

BMI
70
50
30
0
5
10
CREATIN
NAFLD
0
1


COUNTRY
NA
2)Others
1)Born In USA
0.00
0.25
0.50
0.75
1.00
Propensity
NAFLD
0
1

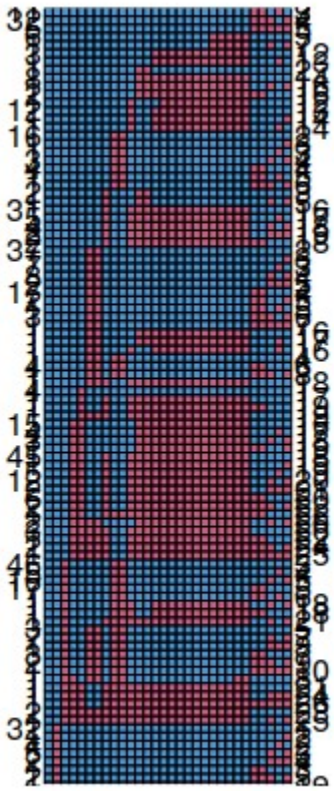

LIVER_TOLD
NA
2)No
1)Yes
NA
2)No
1)Yes
0
1
0
500
1000
1500
2000
2500
Number of Observations


CITIZEN
NA
2)Not a US Citizen
1)US Citizen
6
9
12
15
HBA1C
NAFLD
0
1

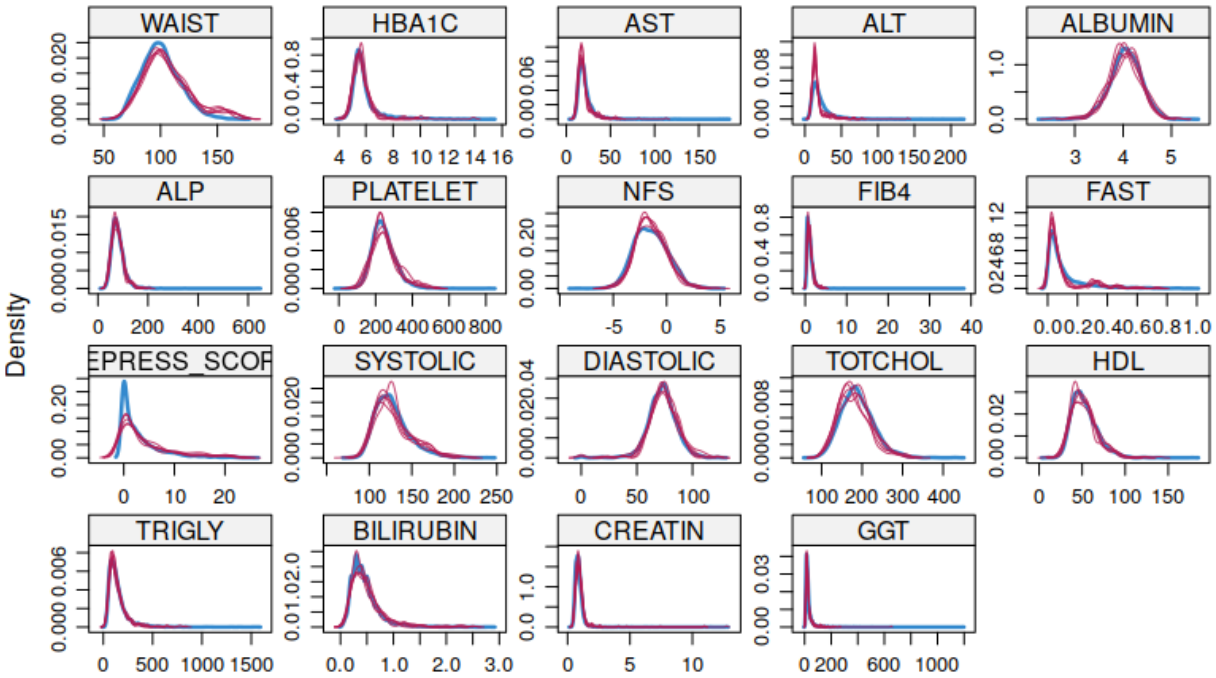

COHORT
2)Study cohort Non-NAFLD
1)Study cohort Presumed NAFLD
2)Study cohort Non-NAFLD
1)Study cohort Presumed NAFLD
0
1
0
1000
2000
Number of Observations

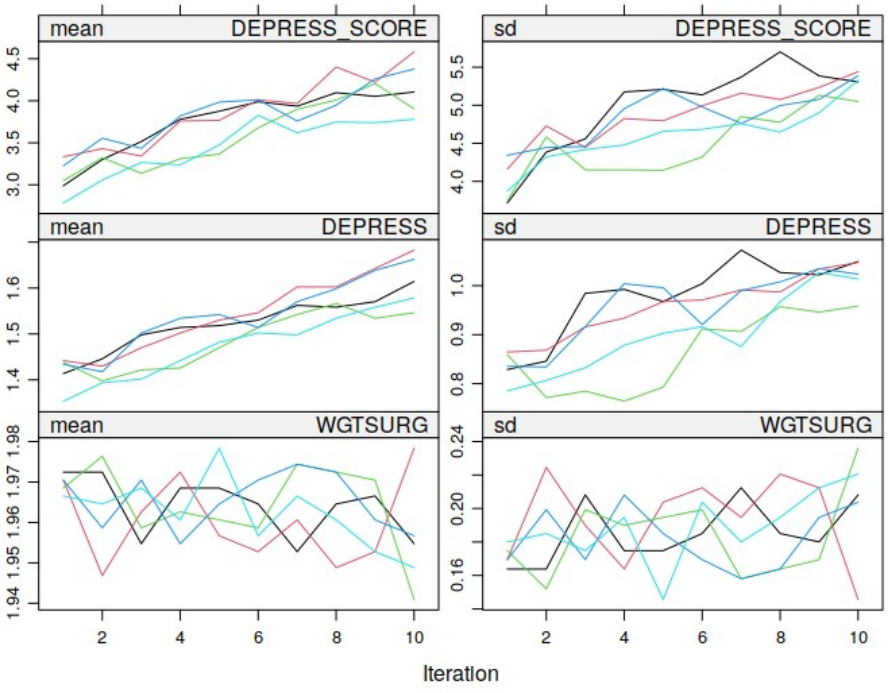

AUTO_HEP
NA
2)No
1)Yes
NA
2)No
1)Yes
0
1
0
1000
2000
Number of Observations

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF
1		H	H	ME	PLA	HB	WAI	ME	ME	SYS	DIAS	ME	TOT	HDL	ALBU	ALT	ALP	TR	BIL	CRE	GG	AST	FAS	ME	NF	FIB	DE	DE	PO	WG	HH	INCOME
2		56	80	114	115	121	123	123	143	149	149	158	182	182	199	200	200	200	200	200	200	207	207	207	208	208	249	249	505	508	547	6189
3	2352	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
4	387	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
5	319	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	2
6	110	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1	2
7	46	1	1	0	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3
8	45	1	1	1	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	18
9	44	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	3
10	37	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2
11	32	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1	0	4
12	32	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
13	31	1	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	16
14	30	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
15	21	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
16	18	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2
17	17	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1	4
18	17	1	1	0	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1	5
19	16	1	1	1	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2
20	16	1	1	0	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	5
21	15	1	1	1	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	16
22	12	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	13
23	12	1	1	1	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	20
24	11	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	1	3
25	8	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	14
26	8	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	4
27	7	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	3



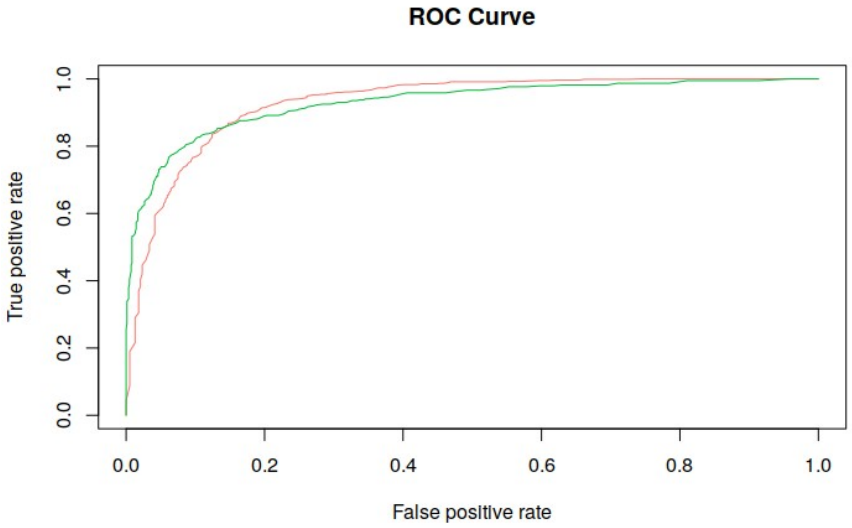
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1		rr.HHINC	rr.POVE	rr.FAST	rr.MET2	rm.HHINC	rm.POVE	rm.FAST	rm.MET2	mr.HHIN	mr.POV	mr.FAST	mr.MET2	mm.HHIN	mm.POVE	mm.FAST	mm.MET2
2	HHINCOME	3284	3282	3119	3119	0	2	165	165	0	44	505	505	547	503	42	42
3	POVERTY	3282	3326	3158	3158	44	0	168	168	2	0	466	466	503	505	39	39
4	FAST	3119	3158	3624	3624	505	466	0	0	165	168	0	0	42	39	207	207
5	MET2	3119	3158	3624	3624	505	466	0	0	165	168	0	0	42	39	207	207
6																	

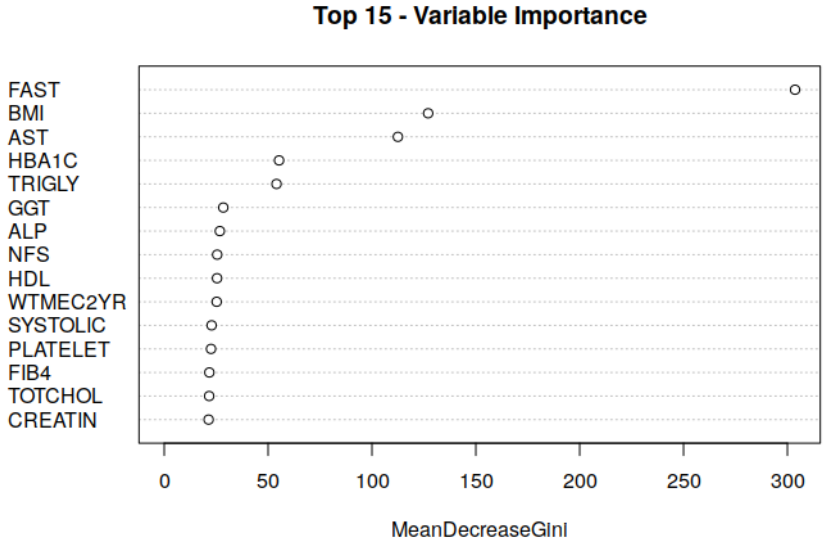


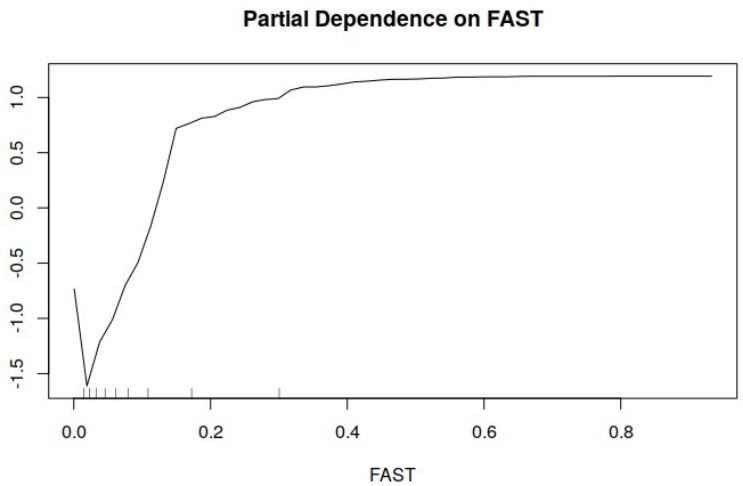


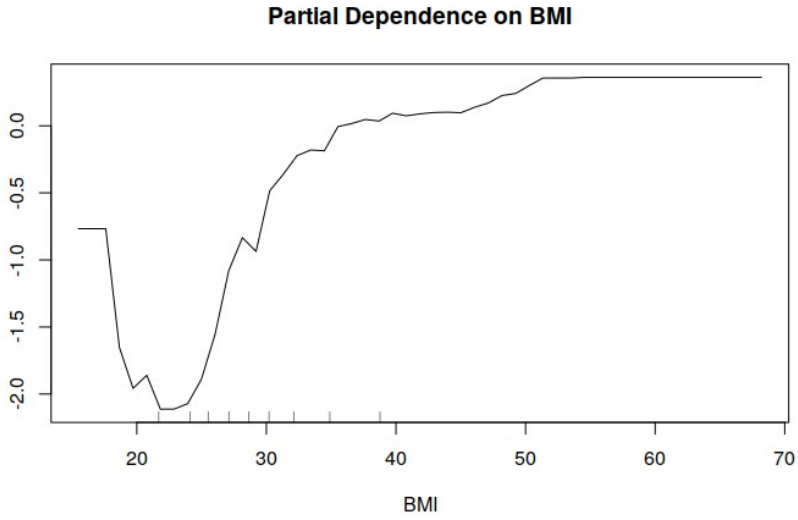
```
> print(rf_default)
Random Forest

2426 samples
  46 predictor
   2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 2183, 2183, 2183, 2183, 2184, 2183, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
   2    0.7815223  0.4375599
  31    0.8524283  0.6465719
  60    0.8594446  0.6636440

Accuracy was used to select the optimal model using the largest value.
```

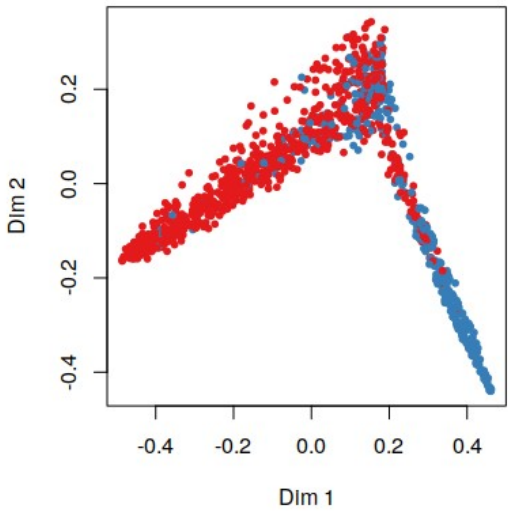
```
> summary(results)

Call:
summary.resamples(object = results)

Models: 50, 100, 150, 200, 250, 300, 350, 400, 450, 500
Number of resamples: 10

Accuracy
         Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
50  0.8073770 0.8328104 0.8553719 0.8470876 0.8618666 0.8806584    0
100 0.8114754 0.8336777 0.8494881 0.8462544 0.8587674 0.8806584    0
150 0.8057851 0.8348808 0.8491736 0.8450113 0.8584711 0.8847737    0
200 0.8032787 0.8317816 0.8533058 0.8470842 0.8644747 0.8847737    0
250 0.8073770 0.8297113 0.8491736 0.8466710 0.8649658 0.8888889    0
300 0.8073770 0.8297113 0.8518324 0.8466727 0.8595041 0.8847737    0
350 0.8073770 0.8276536 0.8512397 0.8487269 0.8665154 0.8971193    0
400 0.8073770 0.8286824 0.8512397 0.8479089 0.8628912 0.8888889    0
450 0.8073770 0.8317816 0.8512397 0.8479072 0.8628912 0.8930041    0
500 0.8073770 0.8326446 0.8536204 0.8483120 0.8595041 0.8930041    0

Kappa
         Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
50  0.5466477 0.5852433 0.6521563 0.6340353 0.6663510 0.7139668    0
100 0.5439438 0.6057619 0.6401099 0.6302995 0.6613184 0.7119794    0
150 0.5146369 0.6087365 0.6357832 0.6273790 0.6597213 0.7228739    0
200 0.5385706 0.6024387 0.6446805 0.6325066 0.6718252 0.7247795    0
250 0.5352906 0.5961688 0.6379818 0.6313626 0.6712010 0.7373183    0
300 0.5439438 0.5968188 0.6422692 0.6316074 0.6601124 0.7266592    0
350 0.5466477 0.5919544 0.6414590 0.6369960 0.6762221 0.7551094    0
400 0.5466477 0.5940544 0.6427802 0.6348221 0.6671226 0.7355182    0
450 0.5466477 0.6024387 0.6427802 0.6352842 0.6694696 0.7461835    0
500 0.5466477 0.6042938 0.6500706 0.6361343 0.6619108 0.7461835    0
```


ROC Curve
True positive rate
False positive rate
0.0
0.2
0.4
0.6
0.8
1.0


Top 15 - Variable Importance
FAST
BMI
AST
HBA1C
TRIGLY
GGT
ALP
NFS
HDL
WTMEC2YR
SYSTOLIC
PLATELET
FIB4
TOTCHOL
CREATIN
0
50
100
150
200
250
300
MeanDecreaseGini


Partial Dependence on FAST
-1.5
-1.0
-0.5
0.0
0.5
1.0
0.0
0.2
0.4
0.6
0.8
FAST


Partial Dependence on BMI
0.0
-0.5
-1.0
-1.5
-2.0
20
30
40
50
60
70
BMI


0.2
0.0
-0.2
-0.4
Dim 2
-0.4
-0.2
0.0
0.2
0.4
Dim 1

CODE

```
#install.packages("mice")
#install.packages("ggplot2")
#install.packages("purrr")
library(mice)
library(ggplot2)
library(purrr)

nhanes <- read.csv(file = 'nhanes.csv')

# Changing NAFLD to two categories 1:having NAFLD, and 0:Non-NAFLD
sum(is.na(nhanes$NAFLD))
nhanes$NAFLD[!nhanes$NAFLD=="1)Non-NAFLD"] <- 1
nhanes$NAFLD[nhanes$NAFLD=="1)Non-NAFLD"] <- 0

# Replacing entries including "Unknown" with NAs

for (i in names(nhanes)) {
  nhanes[,i][grepl("[Uu]nknown", nhanes[,i])] <- NA
}

#Turning character variables into factors

nhanes_char <- nhanes[, sapply(nhanes, class) == 'character']

for(i in names(nhanes_char)){
  nhanes[,i ] <- as.factor(nhanes[,i ])
}

nhanes[ nhanes$FAST>0.3  , c("MET2","FAST")]   # Relation is as I expected
nhanes[ is.na(nhanes$FAST)  , c("MET2","FAST")]   # Relation is as I expected

nhanes[ (nhanes$WAIST >= 102)  & (nhanes$GENDER=="1)Men") ,
c("MET1E","WAIST","GENDER")]
nhanes[ (nhanes$BMI < 25)  , c("MET4","BMI")]  # Relation is as I expected
nhanes[ (nhanes$RIDAGEYR >= 40 & nhanes$RIDAGEYR <= 59  )  , c("RIDAGEYR","AGEGRP")]
# Relation is as I expected
nhanes[ (nhanes$BMI >= 20 ) & (nhanes$BMI < 25 )   , c("BMI","BMIGRP")]  # Relation is as I
expected

ggplot(nhanes, aes(x = ALP, y=MET5)) + geom_point() #Relation is not as I expected.


# Checking correlations
round(cor( nhanes[, sapply(nhanes, class) == 'integer' |sapply(nhanes, class) == 'numeric' ],
     method = c("pearson"), use = "complete.obs"), 3)

boxplot(BMI ~ GENDER, data = nhanes, ylab = "BMI")
```

```
# GLUCOSE and IFG relationship
plot(nhanes$GLUCOSE, nhanes$IFG)
sort(nhanes[nhanes$IFG == 0, "GLUCOSE" ])
sort(nhanes[nhanes$IFG == 1, "GLUCOSE" ])

# Dependence between factor variables
nhanes_fac <- nhanes[, sapply(nhanes, class) == 'factor']
chisq.test(nhanes$STROKE, nhanes$H_CHF, correct=FALSE)

# Treating NAs

# 19615 out of 249015 entries are missing
sum(is.na(nhanes))

# Approximately 8% of the data is missing
mean(is.na(nhanes))

# Distribution of the missing values in data by variables

num_of_nas <- vector()
mean_of_nas <- vector()

num_of_nas <- colSums(is.na(nhanes))
mean_of_nas <- num_of_nas/nrow(nhanes)

mean_of_nas[order(mean_of_nas)]
num_of_nas[order(num_of_nas)]

# Distribution of the missingness by samples

num_of_nas2 <- vector()
mean_of_nas2 <- vector()

num_of_nas2 <- rowSums(is.na(nhanes))
mean_of_nas2 <- num_of_nas2/ncol(nhanes)

mean_of_nas2[order(mean_of_nas2,decreasing = TRUE)]

#To explore md patterns, we exclude variables with less than 50 missing value
vrbls_na_check <- names(num_of_nas) [num_of_nas > 50]

pattern <- md.pattern(nhanes[ , names(nhanes) %in% vrbls_na_check ], rotate.names = TRUE)
write.csv(pattern, file = "pattern.csv")

md_pairs <- md.pairs(nhanes[ , names(nhanes) %in% vrbls_na_check ])
write.csv(md_pairs, file = "md_pairs.csv")

View(md_pairs$rr)
```

```
# We drop variables with more than 45% of missingness

vrbls_to_drop <- names(mean_of_nas) [mean_of_nas > 0.45]
nhanes = nhanes[,!(names(nhanes) %in% vrbls_to_drop)]

#We drop samples with more than 25% missing values
nhanes <- nhanes[  mean_of_nas2 < 0.25,]

# Patterns after dropping the variables and samples
vrbls_na_check <- names(num_of_nas) [num_of_nas > 50]

pattern_after <- md.pattern(nhanes[ , names(nhanes) %in% vrbls_na_check ], rotate.names = TRUE)
write.csv(pattern_after, file = "pattern_after.csv")

md_pairs_after <- md.pairs(nhanes[ , names(nhanes) %in% vrbls_na_check ])
write.csv(md_pairs_after, file = "md_pairs_after.csv")

vrbls_to_drop2 <- c("SEQN" , "COHORT", "AUTO_HEP", "VIR_HEP",
                    "LIV_CIR", "LIV_FIB", "WGTSURG", "BMIGRP", "AGEGRP",
                    "MET2", "MET1E", "MET4", "WAIST")

nhanes = nhanes[,!(names(nhanes) %in% vrbls_to_drop2)]

#imputation with mice
imp <- mice(nhanes,maxit=0)
predM <- imp$predictorMatrix    # I noticed that COHORT is kicked out from the prediction matrix?
Because it is collinear with ??
#predM[ , c("SEQN")]=0   #I got rid of the variable instead
meth <- imp$method
imp <- mice(nhanes,maxit=10)
plot(imp)
nhanes_complete <- complete(imp)
stripplot(imp, NFS~.imp, pch=20, cex=1)
densityplot(imp)


#PLOTS
nhanes_fac <- nhanes[, sapply(nhanes, class) == 'factor']
nhanes_num <- nhanes[, sapply(nhanes, class) == 'numeric']


bar_fun = function(x) {
  ggplot(nhanes, aes(x = .data[[x]]) ) +
    geom_bar(fill = "brown", width = 0.7) +
    coord_flip() +
    xlab(x) + ylab("Number of Observations")
}

bar_plots = map(names(nhanes_fac), ~bar_fun(.x) )
```

```
bar_plots

bar_fill_fun = function(x) {
  ggplot(nhanes, aes(x = .data[[x]]) ) +
    geom_bar() +
    facet_grid(NAFLD ~ .) +
    coord_flip() +
    xlab(x) + ylab("Number of Observations") +
    theme_bw()
}

bar_fill_plots = map(names(nhanes_fac), ~bar_fill_fun(.x) )
bar_fill_plots

bar_fill_prop_fun = function(x) {
  ggplot(nhanes, aes(x = .data[[x]] , fill = NAFLD) ) +
    geom_bar(position = "fill" ) +
    coord_flip() +
    xlab(x) + ylab("Propensity") +
    theme_bw()
}

bar_fill_prop_plots = map(names(nhanes_fac), ~bar_fill_prop_fun(.x) )
bar_fill_prop_plots

scatter_fun = function(x,y) {
  ggplot(nhanes, aes(x = .data[[x]], y= .data[[y]], color=NAFLD)) +
    geom_point() +
    xlab(x) + ylab(y)
}

scatter_plots = map(names(nhanes_num), ~scatter_fun(.x, "BMI") )
scatter_plots

scatter_plots = map(names(nhanes_num), ~scatter_fun(.x, "FAST") )
scatter_plots

box_fun = function(x,y) {
  ggplot(nhanes, aes(x = .data[[x]], y= .data[[y]], color=NAFLD)) +
    geom_boxplot() +
    xlab(x) + ylab(y)
}

box_plots = map(names(nhanes_num), ~box_fun(.x, "CITIZEN") )
box_plots



# Splitting data
```

```
smp_size <- floor(2/3 * nrow(nhanes_complete))

train_ind <- sample(seq_len(nrow(nhanes_complete)), size = smp_size)

#training without "SEQN" , "COHORT", "AUTO_HEP", "VIR_HEP"
#"LIV_CIR", "LIV_FIB", "WGT_SURG", "BMIGRP", "AGEGRP",
#"MET2", "MET1E", "MET4", "WAIST", "LIVER_STILL", "TYPE2DB",
#"DBPREV", "GLUCOSE", "MET1A"

vrbls_to_drop <- c("SEQN" , "COHORT", "AUTO_HEP", "VIR_HEP",
"LIV_CIR", "LIV_FIB", "WGTSURG", "BMIGRP", "AGEGRP",
"MET2", "MET1E", "MET4", "WAIST","LIVER_STILL", "TYPE2DB",
"DBPREV", "GLUCOSE", "MET1A")

train <- nhanes_complete[train_ind, !(names(nhanes) %in% vrbls_to_drop)  ]
test <- nhanes_complete[-train_ind, !(names(nhanes) %in% vrbls_to_drop)  ]


### Random forest
library(randomForest)


library(caret)
library(e1071)

trControl <- trainControl(method = "cv", number = 10, search ="grid")

#46 variables, search with 10 cv

rf_default <- train(NAFLD~.,
                    data = train,
                    method = "rf",
                    metric = "Accuracy",
                    trControl = trControl)
print(rf_default)

plot(rf_default)


#Confusion matrix train
p1_default <- predict(rf_default, train[-3])
confusionMatrix(p1_default, train$NAFLD)

#Confusion matrix test

p1_default <- predict(rf_default, test[-3])
confusionMatrix(p1_default, test$NAFLD)
```

```
#create tunegrid and search with 10 fold cv
tunegrid <- expand.grid(.mtry = 31 )
modellist <- list()

#train with different ntree parameters
for (ntree in c(50,100,150,200,250,300,350,400,450,500)){
  set.seed(123)
  fit <- train(NAFLD~.,
               data = train,
               method = 'rf',
               metric = 'Accuracy',
               tuneGrid = tunegrid,
               trControl = trControl,
               ntree = ntree)
  key <- toString(ntree)
  modellist[[key]] <- fit
}

#Compare results
results <- resamples(modellist)
summary(results)
dotplot(results)

rf <- randomForest( NAFLD ~ . , data = train , mtry=31, proximity=TRUE)
print(rf)
plot(rf)

#Final model ntree=200, mtry=31

rf_final <- randomForest( NAFLD ~ . , data = train , mtry=31,ntree=200, proximity=TRUE)
print(rf_final)
plot(rf_final)

importance(rf_final)

varImpPlot(rf_final,
           sort = T,
           n.var = 15,
           main = "Top 15 - Variable Importance")
partialPlot(rf_final, test, FAST, "1")
partialPlot(rf_final, test, BMI, "1")

MDSplot(rf_final, test$NAFLD)


p <- predict(rf_final, test)
confusionMatrix(p, test$NAFLD)
```

```
# Validation set assessment #2: ROC curves and AUC
# Needs to import ROCR package for ROC curve plotting:
library(ROCR)
# Calculate the probability of new observations belonging to each class
# prediction_for_roc_curve will be a matrix with dimensions data_set_size x number_of_classes
prediction_for_roc_curve <- predict(rf_final,test[,-3],type="prob")
# Use pretty colours:
pretty_colours <- c("#F8766D","#00BA38","#619CFF")
# Specify the different classes
classes <- levels(test$NAFLD)
# For each class
for (i in 1:2)
{
  # Define which observations belong to class[i]
  true_values <- ifelse(test[,3]==classes[i],1,0)
  # Assess the performance of classifier for class[i]
  pred <- prediction(prediction_for_roc_curve[,i],true_values)
  perf <- performance(pred, "tpr", "fpr")
  if (i==1)
  {
    plot(perf,main="ROC Curve",col=pretty_colours[i])
  }
  else
  {
    plot(perf,main="ROC Curve",col=pretty_colours[i],add=TRUE)
  }
  # Calculate the AUC and print it to screen
  auc.perf <- performance(pred, measure = "auc")
  print(auc.perf@y.values)
}
```