Deploying a ML model to predict Fertility outcomes in patient undergoing gamete banking A national SART database

Rstudio

2022-12-01

Introduction: There has been a substantial increase in gamete banking in the last decade, various studies have showed that apart from the scale of this trend, the demographics of women opting for gamete banking has shifted to a younger age group. Success in these cycles is not well defined and the prediction of number of women who eventually returned to use their cryopreserved gametes has been consistently low making an individualized prognosis and counseling using various approaches largely unsuccessful.

Objective: We used a national cohort analysis of 6873 gamete banking cycles undertaken in the USA 2014–2020, to construct a prediction tool to prognosticate patient success rates. The proposed model could assist patient and clinicians throughout all stages of pre-cycle and cycle in a stepwise approach.

Data of interest: Clinical and embryonic data of patients who underwent gamete banking were used. To model the number of oocytes needed to bank and to predict the estimated storage time, we will used logistic regression (model 1) and a stepwise regression (model 2/primary) as methods for selection. The predictive model will be assessed against a test set. The model discrimination and calibration will be tested by ROC curve. Our final model will output the individualized probability based on the predictors. A web-based calculator will be developed to make two types of predictions automatically: 1. Onco-fertility and PGT-M indications and 2. elective Oocyte freezing cycles. This new banking calculator may assist in clinical counseling and in individualized treatment planning and could better address aspects such as the time number of oocytes and cycles required in order to maximize chances for a live birth.

Aims and anticipated outcome: Train test and validate a prediction model based on a national SART dataset. This predictive tool, based on multiple predictors and banking indications will assist in an individual counseling and will produce an estimate of the optimum number of oocytes and cycles needed. Moreover, the tool will estimate the probability and time interval for claiming the gametes.

Methods: A retrospective cohort analysis. Clinical data were extracted at from the national SART database https://www.sart.org/. We included all women undergoing autologous oocyte or embryo banking procedures, who had their first oocyte retrieval from December 2014 to December 2020 and came back to claim their gametes for intended pregnancy.

Cohort description: Our database includes information contained in electronic charts derived most IVF clinic in the USA, from a female population. Data of interest included patient's demographics, past medical history, and infertility evaluation including diagnosis, laboratory testing for ovarian reserve, and any radiologic studies pertinent to a diagnosis of infertility.

Definition of outcomes: We aimed to predict live birth among those who came to claim their gametes (task 1). The response variable "live birth" is coded 0 for no live birth and 1 for live birth.

Upload data

```
library(readr)
bank.linked_df <- read.csv("C:\\Users\\Youval Fouks\\Desktop\\Practicum\\CSV\\Linked 2022 SART Data.csv
```

Install & load packages

```r
if (!require(Hmisc)) {install.packages("Hmisc")}
```

```
## Loading required package: Hmisc

## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

## Loading required package: ggplot2

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```r
if (!require(MASS)) {install.packages("MASS")}
```

```
## Loading required package: MASS
```

```r
if (!require(caret)) {install.packages("caret")}
```

```
## Loading required package: caret

##
## Attaching package: 'caret'

## The following object is masked from 'package:survival':
##
##     cluster
```

```r
if (!require(leaps)) {install.packages("leaps")}
```

```
## Loading required package: leaps
```

```r
if (!require(gamlr)) {install.packages("gamlr")}
```

```
## Loading required package: gamlr

## Loading required package: Matrix
```

```r
if (!require(glmnet)) {install.packages("glmnet")}
```

```
## Loading required package: glmnet

## Loaded glmnet 4.1-6
```

```r
if (!require(sas7bdat)) {install.packages("sas7bdat")}
```

```
## Loading required package: sas7bdat
```

```r
if (!require(dplyr)) {install.packages("dplyr")}
```

```
## Loading required package: dplyr

##
## Attaching package: 'dplyr'

## The following object is masked from 'package:MASS':
##
##     select

## The following objects are masked from 'package:Hmisc':
##
##     src, summarize

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
if (!require(purrr)) {install.packages("purrr")}
```

```
## Loading required package: purrr

##
## Attaching package: 'purrr'

## The following object is masked from 'package:caret':
##
##     lift
```

```r
if (!require(pROC)) {install.packages("pROC")}
```

```
## Loading required package: pROC

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
if (!require(survivalROC)) {install.packages("survivalROC")}
```

```
## Loading required package: survivalROC
```

```r
if (!require(survival)) {install.packages("survival")}
if (!require(tidyr)) {install.packages("tidyr")}
```

```
## Loading required package: tidyr

##
## Attaching package: 'tidyr'

## The following objects are masked from 'package:Matrix':
##
##     expand, pack, unpack
```

```r
if (!require(ggplot2)) {install.packages("ggplot2")}
if (!require(arsenal)) {install.packages("arsenal")}
```

```
## Loading required package: arsenal

##
## Attaching package: 'arsenal'

## The following object is masked from 'package:Hmisc':
##
##     %nin%
```

```r
if (!require(tidyverse)) {install.packages("tidyverse")}
```

```
## Loading required package: tidyverse

## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v tibble  3.1.8     v forcats 0.5.2
## v stringr 1.4.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x tidyr::expand()    masks Matrix::expand()
## x dplyr::filter()    masks stats::filter()
## x dplyr::lag()       masks stats::lag()
## x purrr::lift()      masks caret::lift()
## x tidyr::pack()      masks Matrix::pack()
## x dplyr::select()    masks MASS::select()
## x dplyr::src()       masks Hmisc::src()
## x dplyr::summarize() masks Hmisc::summarize()
## x tidyr::unpack()    masks Matrix::unpack()
```

```r
if (!require(mice)) {install.packages("mice")}
```

```
## Loading required package: mice
##
## Attaching package: 'mice'
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
library(naniar)
```

```
df <- bank.linked_df
```

```
bank.linked_df$PatientAgeAtStart <- as.numeric(bank.linked_df$PatientAgeAtStart)
```

Preprocessing: Redundancy of the data was treated with variable Selection in order to reach better predictors for the modeling phase. Redundant and highly correlated variables exclusion was made (by clinical judgment).

Remove unnecessary cols

```
df <- subset(df, select = -c(5, 6:15, 16:19, 49:52, 56:65, 69, 73:75, 79:81, 87, 88, 93, 100, 101, 103,
```

rename variables

```
names(df)[5]=paste("Age")
```

recode var and recode to missing data

```
df[df == "N"] <- "0"
df[df == "Y"] <- "1"
df[df == "No"] <- "0"
df[df == "Yes"] <- "1"

# recode to missing data
df[df=="NaN"] <- NA
df[df=="N/A"] <- NA
df[df=="Not Reported"] <- NA
df[df == "Not Entered"] <- NA
df[df == ""] <- NA
df[df == "NULL"] <- NA
```

Missingness in data

```
naniar::miss_var_summary(df)
```

```
## # A tibble: 124 x 3
##    variable                                                n_miss pct_m~1
```

```
##    <chr>                                                    <int>   <dbl>
##  1 gender.dysphoria                                          6872     100
##  2 DO                                                        6872     100
##  3 OocyteThawDateStartDateDiff_Until2013Only                 6872     100
##  4 ThawedOocyteRetrievalToCryoPreservationDateDiff_Until2013Only    6872     100
##  5 ThawedOocyteDonorRetrievalToCryoPreservationDateDiff_Until201~   6872     100
##  6 StartDateFreezeDateDiff_AutologousRetrieval2_Freeze3_3_Autolo~   6872     100
##  7 StartDateFreezeDateDiff_AutologousRetrieval4_Freeze3_3_Autolo~   6872     100
##  8 RetrievalDateStartDateDiff_AutologousRetrieval5           6872     100
##  9 StartDateFreezeDateDiff_AutologousRetrieval5_Freeze2_2_Autolo~   6872     100
## 10 StartDateFreezeDateDiff_AutologousRetrieval5_Freeze3_3_Autolo~   6872     100
## # ... with 114 more rows, and abbreviated variable name 1: pct_miss
```

```
df <- df[,colSums(is.na(df))<nrow(df)]
naniar::miss_var_summary(df)
```

```
## # A tibble: 109 x 3
##    variable                                                n_miss pct_m~1
##    <chr>                                                    <int>   <dbl>
##  1 StartDateFreezeDateDiff_AutologousRetrieval3_Freeze3_3_Autolo~   6871   100.
##  2 StartDateFreezeDateDiff_AutologousRetrieval4_Freeze2_2_Autolo~   6871   100.
##  3 StartDateFreezeDateDiff_AutologousRetrieval3_Freeze2_2_Autolo~   6870   100.
##  4 Medical.Tumor.prevention                                  6869   100.
##  5 StartDateFreezeDateDiff_AutologousRetrieval6              6869   100.
##  6 TransferDateRetrievalDateDiff_AutologousRetrieval6        6869   100.
##  7 RetrievalDateStartDateDiff_AutologousRetrieval4           6868    99.9
##  8 OocytesCryoed_AutologousRetrieval4                        6868    99.9
##  9 StartDateFreezeDateDiff_AutologousRetrieval2_Freeze2_2_Autolo~   6867    99.9
## 10 StartDateFreezeDateDiff_AutologousRetrieval6_Freeze1_1_Autolo~   6867    99.9
## # ... with 99 more rows, and abbreviated variable name 1: pct_miss
```

Remove splitted cycles

```
df_no_fresh <- df[!(df$RetrievalType_AutologousRetrieval1=="Fresh"),]
df <- df_no_fresh
```

Table descriptive

```
Table1 <- tableby(PregnancyOutcome_Live.Birth ~ Age + Clinic.Region.USA + Gravidity +  FullTermBirths +
```

```
Table2 <- tableby(C ~ ThawedEmbryo + ThawedOocyte + SpermSource_Partner + SpermSource_Donor + SpermSour

##########unknowen and 2PN excluded
#summary(tab1, text=TRUE)
summary(Table1)
```

```
##
##
## |                              |   0 (N=4192)    |   1 (N=2346)    | Total (N=6538)  | p value|
## |:-----------------------------|:---------------:|:---------------:|:---------------:|-------:|
## |**Age**                       |                 |                 |                 | < 0.001|
```

```
## |   Mean (SD)    | 35.822 (5.137)  | 33.745 (4.620)  | 35.077 (5.056)  |         |
## |   Range        | 22.000 - 53.000 | 18.000 - 48.000 | 18.000 - 53.000 |         |
## |**Clinic.Region.USA**         |                 |                 |                 | < 0.001 |
## |   Midwest      |   726 (17.3%)   |   378 (16.1%)   |  1104 (16.9%)   |         |
## |   Northeast    |  1143 (27.3%)   |   483 (20.6%)   |  1626 (24.9%)   |         |
## |   South        |  1519 (36.2%)   |  1001 (42.7%)   |  2520 (38.5%)   |         |
## |   West         |   804 (19.2%)   |   484 (20.6%)   |  1288 (19.7%)   |         |
## |**Gravidity**                 |                 |                 |                 |   0.213 |
## |   >10          |    1 (0.0%)     |    0 (0.0%)     |    1 (0.0%)     |         |
## |   0            |  2024 (48.3%)   |  1110 (47.3%)   |  3134 (47.9%)   |         |
## |   1            |  1051 (25.1%)   |   628 (26.8%)   |  1679 (25.7%)   |         |
## |   10           |    1 (0.0%)     |    1 (0.0%)     |    2 (0.0%)     |         |
## |   2            |   566 (13.5%)   |   333 (14.2%)   |   899 (13.8%)   |         |
## |   3            |   293 (7.0%)    |   156 (6.6%)    |   449 (6.9%)    |         |
## |   4            |   133 (3.2%)    |    70 (3.0%)    |   203 (3.1%)    |         |
## |   5            |    55 (1.3%)    |    28 (1.2%)    |    83 (1.3%)    |         |
## |   6            |    39 (0.9%)    |    9 (0.4%)     |    48 (0.7%)    |         |
## |   7            |    15 (0.4%)    |    3 (0.1%)     |    18 (0.3%)    |         |
## |   8            |    4 (0.1%)     |    5 (0.2%)     |    9 (0.1%)     |         |
## |   9            |    1 (0.0%)     |    0 (0.0%)     |    1 (0.0%)     |         |
## |   Unknown      |    9 (0.2%)     |    3 (0.1%)     |    12 (0.2%)    |         |
## |**FullTermBirths**            |                 |                 |                 |   0.002 |
## |   N-Miss       |      2033       |      1113       |      3146       |         |
## |   0            |  1084 (50.2%)   |   535 (43.4%)   |  1619 (47.7%)   |         |
## |   1            |   825 (38.2%)   |   547 (44.4%)   |  1372 (40.4%)   |         |
## |   2            |   167 (7.7%)    |   113 (9.2%)    |   280 (8.3%)    |         |
## |   3            |    50 (2.3%)    |    30 (2.4%)    |    80 (2.4%)    |         |
## |   4            |    25 (1.2%)    |    7 (0.6%)     |    32 (0.9%)    |         |
## |   5            |    6 (0.3%)     |    1 (0.1%)     |    7 (0.2%)     |         |
## |   7            |    1 (0.0%)     |    0 (0.0%)     |    1 (0.0%)     |         |
## |   8            |    1 (0.0%)     |    0 (0.0%)     |    1 (0.0%)     |         |
## |**PreTermBirths**             |                 |                 |                 |   0.145 |
## |   N-Miss       |      2033       |      1113       |      3146       |         |
## |   0            |  1976 (91.5%)   |  1146 (92.9%)   |  3122 (92.0%)   |         |
## |   1            |   157 (7.3%)    |    83 (6.7%)    |   240 (7.1%)    |         |
## |   2            |    18 (0.8%)    |    4 (0.3%)     |    22 (0.6%)    |         |
## |   3            |    3 (0.1%)     |    0 (0.0%)     |    3 (0.1%)     |         |
## |   4            |    3 (0.1%)     |    0 (0.0%)     |    3 (0.1%)     |         |
## |   Unknown      |    2 (0.1%)     |    0 (0.0%)     |    2 (0.1%)     |         |
## |**MaleInfertility**           |                 |                 |                 | < 0.001 |
## |   0            |  2577 (61.5%)   |  1338 (57.0%)   |  3915 (59.9%)   |         |
## |   1            |  1615 (38.5%)   |  1008 (43.0%)   |  2623 (40.1%)   |         |
## |**Endometriosis**             |                 |                 |                 |   0.299 |
## |   0            |  3908 (93.2%)   |  2171 (92.5%)   |  6079 (93.0%)   |         |
## |   1            |   284 (6.8%)    |   175 (7.5%)    |   459 (7.0%)    |         |
## |**PolycysticOvaries**         |                 |                 |                 |   0.004 |
## |   0            |  3962 (94.5%)   |  2175 (92.7%)   |  6137 (93.9%)   |         |
## |   1            |   230 (5.5%)    |   171 (7.3%)    |   401 (6.1%)    |         |
## |**DiminishedOvarianReserve**  |                 |                 |                 | < 0.001 |
## |   0            |  3205 (76.5%)   |  2054 (87.6%)   |  5259 (80.4%)   |         |
## |   1            |   987 (23.5%)   |   292 (12.4%)   |  1279 (19.6%)   |         |
## |**TubalLigation**             |                 |                 |                 |   0.784 |
## |   0            |  4135 (98.6%)   |  2316 (98.7%)   |  6451 (98.7%)   |         |
## |   1            |    57 (1.4%)    |    30 (1.3%)    |    87 (1.3%)    |         |
```

```
## |**Uterine**                    |                |                |                |   0.543|
## |   0            |  3971 (94.7%)  |  2214 (94.4%)  |  6185 (94.6%)  |        |
## |   1            |   221 (5.3%)   |   132 (5.6%)   |   353 (5.4%)   |        |
## |**Unexplained**                |                |                |                |   0.012|
## |   0            |  3698 (88.2%)  |  2019 (86.1%)  |  5717 (87.4%)  |        |
## |   1            |   494 (11.8%)  |   327 (13.9%)  |   821 (12.6%)  |        |
## |**as.factor(elect)**           |                |                |                | < 0.001|
## |   N-Miss       |       505      |       336      |       841      |        |
## |   1            |  3687 (100.0%) |  2010 (100.0%) |  5697 (100.0%) |        |
## |**as.factor(Onco)**            |                |                |                |   0.002|
## |   N-Miss       |      4158      |      2333      |      6491      |        |
## |   1            |   34 (100.0%)  |   13 (100.0%)  |   47 (100.0%)  |        |
## |**as.factor(Medical.benign)**  |                |                |                | < 0.001|
## |   N-Miss       |      3848      |      2141      |      5989      |        |
## |   1            |  344 (100.0%)  |  205 (100.0%)  |  549 (100.0%)  |        |
## |**as.factor(Syndromatic.DOR)** |                |                |                |   0.003|
## |   N-Miss       |      4174      |      2342      |      6516      |        |
## |   1            |   18 (100.0%)  |   4 (100.0%)   |   22 (100.0%)  |        |
## |**as.factor(DOR..35)**         |                |                |                | < 0.001|
## |   N-Miss       |      4043      |      2271      |      6314      |        |
## |   1            |  149 (100.0%)  |   75 (100.0%)  |  224 (100.0%)  |        |
```

```
summary(Table2)
```

```
##
##
## |                                                  |  0 (N=2344)   |  1 (N=4194)   | Total (N=6538
## |:-------------------------------------------------|:-------------:|:-------------:|:-------------
## |**ThawedEmbryo**                                  |               |               |
## |   0                               |  605 (25.8%)  | 2116 (50.5%)  |  2721 (41.6%)
## |   1                               | 1739 (74.2%)  | 2078 (49.5%)  |  3817 (58.4%)
## |**ThawedOocyte**                                  |               |               |
## |   0                               | 1727 (73.7%)  | 2048 (48.8%)  |  3775 (57.7%)
## |   1                               |  617 (26.3%)  | 2146 (51.2%)  |  2763 (42.3%)
## |**SpermSource_Partner**                           |               |               |
## |   0                               | 1884 (80.4%)  | 2501 (59.6%)  |  4385 (67.1%)
## |   1                               |  460 (19.6%)  | 1693 (40.4%)  |  2153 (32.9%)
## |**SpermSource_Donor**                             |               |               |
## |   0                               | 2190 (93.4%)  | 3752 (89.5%)  |  5942 (90.9%)
## |   1                               |  154 (6.6%)   |  442 (10.5%)  |   596 (9.1%)
## |**SpermSource_Mixed**                             |               |               |
## |   0                               | 2342 (99.9%)  | 4186 (99.8%)  |  6528 (99.8%)
## |   1                               |   2 (0.1%)    |   8 (0.2%)    |   10 (0.2%)
## |**TransferAttempted**                             |               |               |
## |   0                               |   3 (0.1%)    |  873 (20.8%)  |  876 (13.4%)
## |   1                               | 2341 (99.9%)  | 3321 (79.2%)  |  5662 (86.6%)
## |**TreatmentOutcome_Not.Pregnant**                 |               |               |
## |   0                               | 2344 (100.0%) | 1161 (27.7%)  |  3505 (53.6%)
## |   1                               |   0 (0.0%)    | 3033 (72.3%)  |  3033 (46.4%)
## |**TreatmentOutcome_Biochemical**                  |               |               |
## |   0                               | 2344 (100.0%) | 3611 (86.1%)  |  5955 (91.1%)
## |   1                               |   0 (0.0%)    |  583 (13.9%)  |   583 (8.9%)
## |**TreatmentOutcome_Clinical.Intrauterine.Gestation** |            |               |
## |   0                               |   1 (0.0%)    | 3645 (86.9%)  |  3646 (55.8%)
```

```
## |   1                                  | 2343 (100.0%) |  549 (13.1%)  |  2892 (44.2%)
## |**PregnancyOutcome_Outcome.Unknown**                |               |               |
## |   0                                  | 2344 (100.0%) | 4178 (99.6%)  |  6522 (99.8%)
## |   1                                  |   0 (0.0%)    |   16 (0.4%)   |   16 (0.2%)
## |**PregnancyOutcome_Live.Birth**                     |               |               |
## |   0                                  |   0 (0.0%)    | 4192 (100.0%) |  4192 (64.1%)
## |   1                                  | 2344 (100.0%) |   2 (0.0%)    |  2346 (35.9%)
## |**Pregnancy.Loss.Abortion**                         |               |               |
## |   0                                  | 2344 (100.0%) | 3676 (87.6%)  |  6020 (92.1%)
## |   1                                  |   0 (0.0%)    |  518 (12.4%)  |   518 (7.9%)
```

Sum retrival cycles into sum

```
df$Retrieval_1 <-ifelse(df$ThawDateStartDateDiff_AutologousRetrieval1 == "NA" , 0, 1)
df$Retrieval_2 <-ifelse(df$ThawDateStartDateDiff_AutologousRetrieval2 == "NA" , 0, 1)
df$Retrieval_3 <-ifelse(df$ThawDateStartDateDiff_AutologousRetrieval3 == "NA" , 0, 1)
df$Retrieval_4 <-ifelse(df$ThawDateStartDateDiff_AutologousRetrieval4 == "NA" , 0, 1)
df$Retrieval_5 <-ifelse(df$ThawDateStartDateDiff_AutologousRetrieval5 == "NA" , 0, 1)
df$Retrieval_6 <-ifelse(df$ThawDateStartDateDiff_AutologousRetrieval6 == "NA" , 0, 1)

df <- df %>%
  mutate(sum_1 = rowSums(across(c(Retrieval_1, Retrieval_2, Retrieval_3, Retrieval_4, Retrieval_5, Retri
```

```
df <- subset(df, select = -c(110:115))
```

Number of retrivels against the results

```
Table3 <- tableby(C ~ as.factor(sum_1) ,data=df)

##########unknowen and 2PN excluded
#summary(tab1, text=TRUE)
summary(Table3)
```

```
##
##
## |                     |  0 (N=2344)  |  1 (N=4194)  | Total (N=6538) | p value|
## |:--------------------|:------------:|:------------:|:--------------:|-------:|
## |**as.factor(sum_1)** |              |              |                | < 0.001|
## |   1  | 2247 (95.9%) | 3853 (91.9%) |  6100 (93.3%)  |        |
## |   2  |  82 (3.5%)   |  258 (6.2%)  |   340 (5.2%)   |        |
## |   3  |  13 (0.6%)   |  63 (1.5%)   |   76 (1.2%)    |        |
## |   4  |   2 (0.1%)   |  10 (0.2%)   |   12 (0.2%)    |        |
## |   5  |   0 (0.0%)   |   4 (0.1%)   |    4 (0.1%)    |        |
## |   6  |   0 (0.0%)   |   6 (0.1%)   |    6 (0.1%)    |        |
```

```
df$Time_to.claim_1 <- ifelse(df$TransferDateRetrievalDateDiff_AutologousRetrieval1 == "NA" , "NA", df$T
df$Time_to.claim_2 <- ifelse(df$TransferDateRetrievalDateDiff_AutologousRetrieval2 == "NA" , "NA", df$T
df$Time_to.claim_3 <- ifelse(df$TransferDateRetrievalDateDiff_AutologousRetrieval3 == "NA" , "NA", df$T
df$Time_to.claim_4 <- ifelse(df$TransferDateRetrievalDateDiff_AutologousRetrieval4 == "NA" , "NA", df$T
df$Time_to.claim_5 <- ifelse(df$TransferDateRetrievalDateDiff_AutologousRetrieval5 == "NA" , "NA", df$T
df$Time_to.claim_6 <- ifelse(df$TransferDateRetrievalDateDiff_AutologousRetrieval6 == "NA" , "NA", df$T
```

```
df$Time_to.claim_1 <- as.numeric(df$Time_to.claim_1)
df$Time_to.claim_2 <- as.numeric(df$Time_to.claim_2)
df$Time_to.claim_3 <- as.numeric(df$Time_to.claim_3)
df$Time_to.claim_4 <- as.numeric(df$Time_to.claim_4)
df$Time_to.claim_5 <- as.numeric(df$Time_to.claim_5)
df$Time_to.claim_6 <- as.numeric(df$Time_to.claim_6)
```

```
df <- df %>%
  mutate(sum_2 = rowSums(across(c(Time_to.claim_1, Time_to.claim_2, Time_to.claim_3, Time_to.claim_4, T

names(df)[111]=paste("Time_to.claim")
```

```
df <- subset(df, select = -c(112:116))
#df <- subset(df, select = -c(112))
```

```
names(df)[110]=paste("total_n_fresh_cycles")
names(df)[112]=paste("total_tolast_Thaw")
```

Trim all in between cycles after aggregating the data

```
df.big.trim <- subset(df, select = -c(58:106))
#df <- subset(df, select = -c(112))
```

```
naniar::miss_var_summary(df.big.trim)
```

```
## # A tibble: 63 x 3
##    variable                                        n_miss pct_miss
##    <chr>                                            <int>    <dbl>
##  1 RetrievalDateStartDateDiff_AutologousRetrieval1   6538    100
##  2 Medical.Tumor.prevention                          6535    100.
##  3 Syndromatic.DOR                                   6516     99.7
##  4 Onco                                              6491     99.3
##  5 DOR..35                                           6314     96.6
##  6 OtherReasonRFA                                    6037     92.3
##  7 Medical.benign                                    5989     91.6
##  8 Unknowen                                          5930     90.7
##  9 StartDateFreezeDateDiff_AutologousRetrieval1      4933     75.5
## 10 NumberBorn                                        4179     63.9
## # ... with 53 more rows
```

Missing values: I have been faced with missing data of various extent. After removing structural missingness, we have summed our missing percentages that in some cases has reached 11%.

Treatment for missing data: First: for each attribute, I test for the correlation of missingness with the outcome and explore with the clinician (my boss) whether or not missingness could potentially encode for a certain prognosis. After solving all structural missingness, I have checked for relations between target and potentially imputed variables (scatter kNN). Using kNN-imputation (k=20) for treating missing data and to control noise in the data by assigning values to miss data based on the closest class or cluster that does not have a missing value.

Treat structural missingness

```
df.big.trim$FullTermBirths <- ifelse(df.big.trim$Gravidity == 0 , 0, df.big.trim$FullTermBirths)
df.big.trim$PreTermBirths <- ifelse(df.big.trim$Gravidity == 0 , 0, df.big.trim$PreTermBirths)
df.big.trim$BiochemicalPregnancies <- ifelse(df.big.trim$Gravidity == 0 , 0, df.big.trim$BiochemicalPreg
df.big.trim$SpontaneousAbortions <- ifelse(df.big.trim$Gravidity == 0 , 0, df.big.trim$SpontaneousAborti

df.big.trim$NumberBorn[is.na(df.big.trim$NumberBorn)] <- 0
df.big.trim$NumberLiveBorn[is.na(df.big.trim$NumberLiveBorn)] <- 0

df.big.trim$Unknowen[is.na(df.big.trim$Unknowen)] <- 0
df.big.trim$elect[is.na(df.big.trim$elect)] <- 0
df.big.trim$Onco[is.na(df.big.trim$Onco)] <- 0
df.big.trim$Medical.benign[is.na(df.big.trim$Medical.benign)] <- 0
df.big.trim$Medical.Tumor.prevention[is.na(df.big.trim$Medical.Tumor.prevention)] <- 0
df.big.trim$Syndromatic.DOR[is.na(df.big.trim$Syndromatic.DOR)] <- 0
df.big.trim$DOR..35[is.na(df.big.trim$DOR..35)] <- 0
```

```
df.big.trim <- subset(df.big.trim, select = -c(9, 24, 38, 44, 45, 51, 54, 57,59))
```

```
df <-df.big.trim

df$TransferAttempted <- as.factor(df$TransferAttempted)
df$Clinic.Region.USA <- as.factor(df$Clinic.Region.USA)
df$Age <- as.numeric(df$Age)
df$RetrievalType_AutologousRetrieval1.1 <- as.numeric(df$RetrievalType_AutologousRetrieval1.1)
```

```
## Warning: NAs introduced by coercion
```

```
df$NumberBorn <- as.numeric(df$NumberBorn)
df$DonorIdentityKnown  <- as.factor(df$DonorIdentityKnown )
df$Total2PN <- as.numeric(df$Total2PN)

cols <- names(df)[7:10] # or column index (change the index if needed)
df[cols] <- lapply(df[cols], as.integer)
```

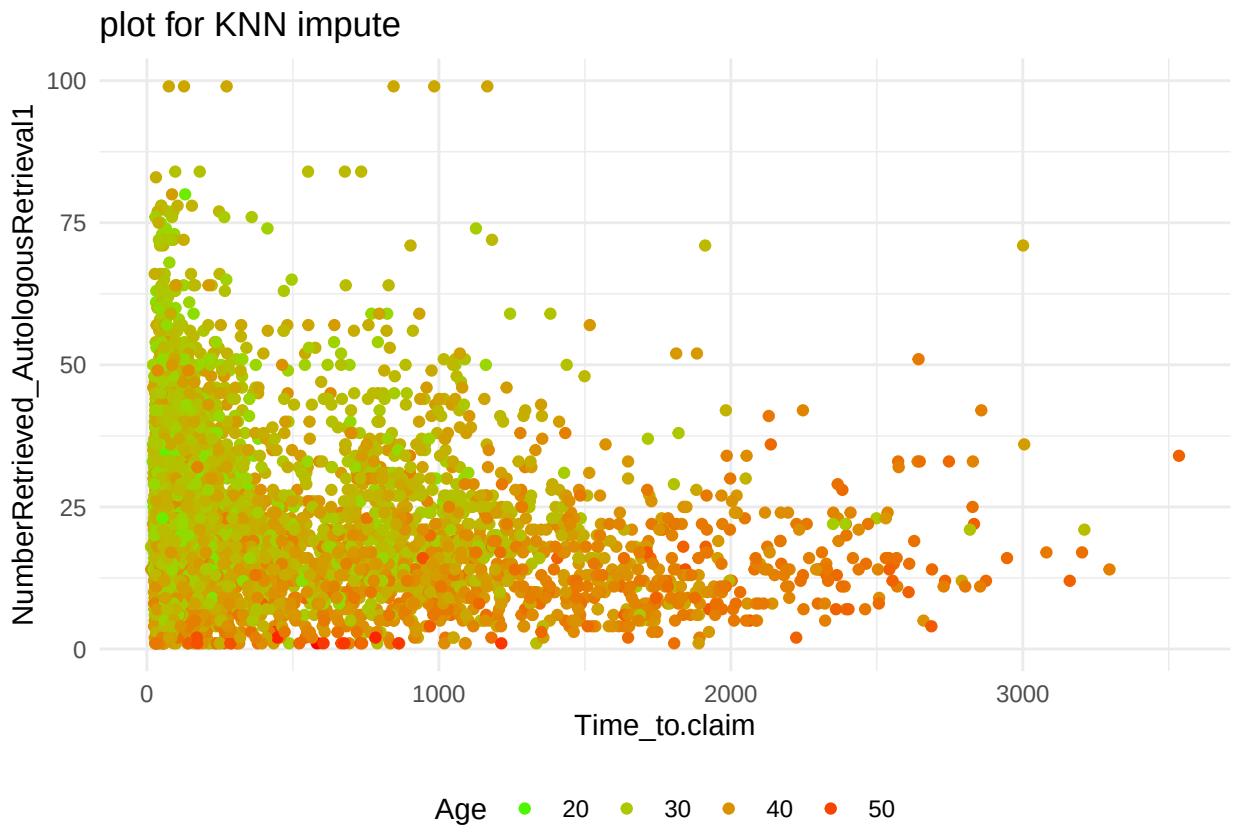
```
## Warning in lapply(df[cols], as.integer): NAs introduced by coercion

## Warning in lapply(df[cols], as.integer): NAs introduced by coercion

## Warning in lapply(df[cols], as.integer): NAs introduced by coercion
```

```
cols1 <- names(df)[46:49] # or column index (change the index if needed)
df[cols1] <- lapply(df[cols1], as.numeric)
cols2 <- names(df)[1:4] # or column index (change the index if needed)
df[cols2] <- lapply(df[cols2], as.factor)
cols3 <- names(df)[11:35] # or column index (change the index if needed)
df[cols3] <- lapply(df[cols3], as.factor)
cols4 <- names(df)[38:45] # or column index (change the index if needed)
df[cols4] <- lapply(df[cols4], as.factor)

#trim missing
trim <- c("RetrievalType_AutologousRetrieval1.1", "Total2PN","StartDateFreezeDateDiff_AutologousRetrieva
df <- df[, !(names(df) %in% trim)]
```

kNN Imputation for Missing Values Relations between target and imputed

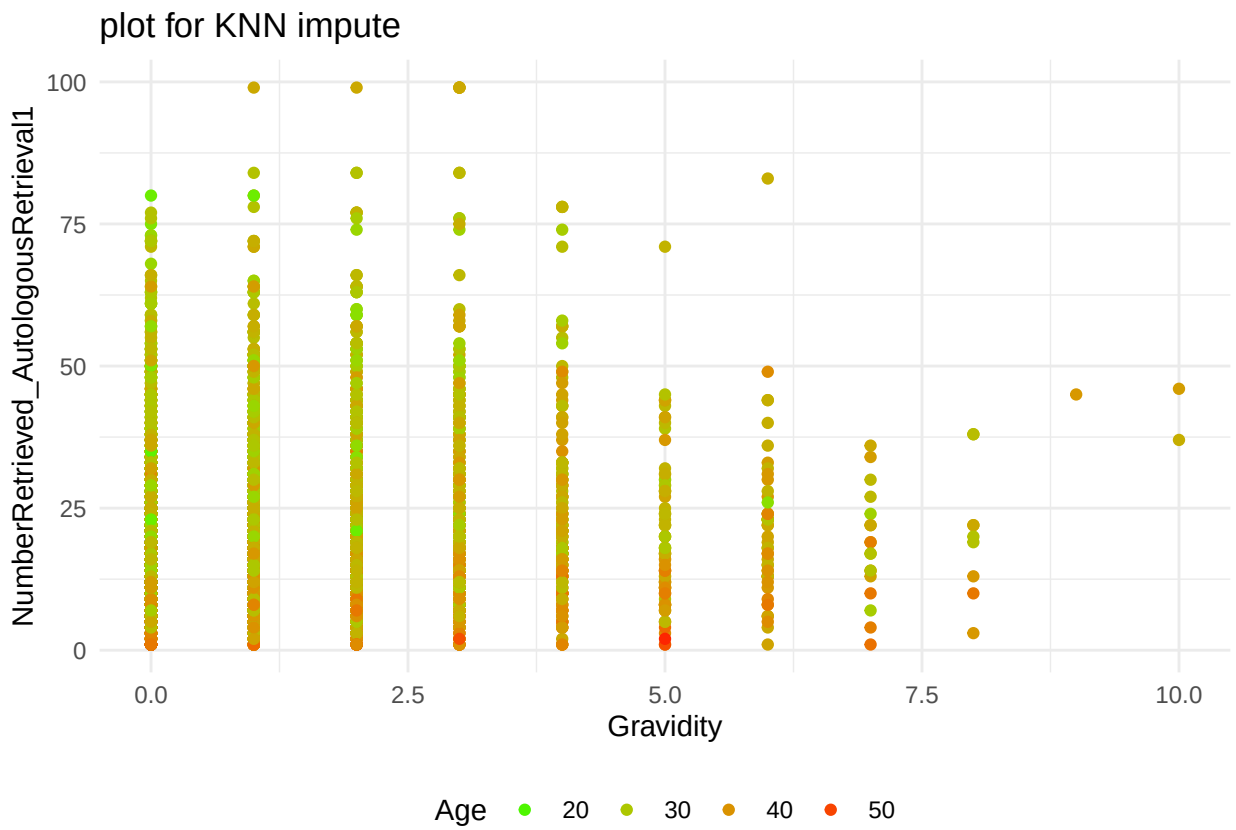
```
ggplot(df, aes(x = Time_to.claim , y = NumberRetrieved_AutologousRetrieval1, color = Age)) +
  geom_point(show.legend = TRUE) +
  labs(x = 'Time_to.claim', y='NumberRetrieved_AutologousRetrieval1',  title = "plot for KNN impute",
       color = 'Age') +
  scale_color_gradient(low = "green", high = "red",
                       na.value = "blue", guide = "legend") +
  theme_minimal()+theme(legend.position="bottom")
```

```
## Warning: Removed 876 rows containing missing values (‘geom_point()‘).
```



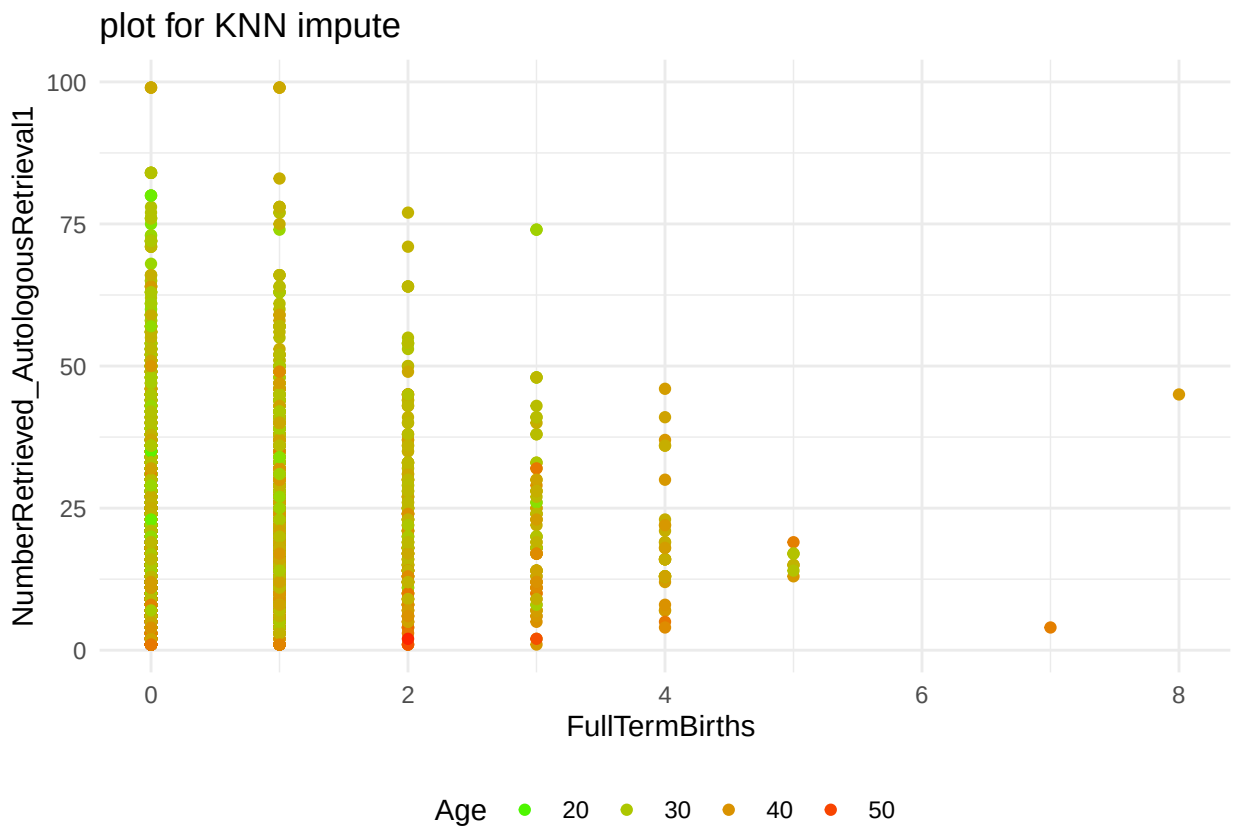

```
ggplot(df, aes(x = Gravidity , y = NumberRetrieved_AutologousRetrieval1, color = Age)) +
  geom_point(show.legend = TRUE) +
  labs(x = 'Gravidity', y='NumberRetrieved_AutologousRetrieval1',  title = "plot for KNN impute",
       color = 'Age') +
  scale_color_gradient(low = "green", high = "red",
                       na.value = "blue", guide = "legend") +
  theme_minimal()+theme(legend.position="bottom")
```

```
## Warning: Removed 13 rows containing missing values (‘geom_point()‘).
```




```
ggplot(df, aes(x = FullTermBirths  , y = NumberRetrieved_AutologousRetrieval1, color = Age)) +
  geom_point(show.legend = TRUE) +
  labs(x = 'FullTermBirths ', y='NumberRetrieved_AutologousRetrieval1',  title = "plot for KNN impute",
       color = 'Age') +
  scale_color_gradient(low = "green", high = "red",
                       na.value = "blue", guide = "legend") +
  theme_minimal()+theme(legend.position="bottom")
```

```
## Warning: Removed 12 rows containing missing values (‘geom_point()‘).
```



kNN Imputation

```
library(caret)
library(RANN)
preProcValues <- preProcess(df %>%
                            dplyr::select(Time_to.claim, Gravidity, FullTermBirths ,Age,  DonorIdentityKn
                              method = c("knnImpute"),
                              k = 20,
                              knnSummary = mean)
impute_df_info <- predict(preProcValues, df,na.action = na.pass)

procNames <- data.frame(col = names(preProcValues$mean), mean = preProcValues$mean, sd = preProcValues$s
for(i in procNames$col){
 impute_df_info[i] <- impute_df_info[i]*preProcValues$std[i]+preProcValues$mean[i]
}
```

```
df <- impute_df_info
pct_miss(df)
```

```
## [1] 0
```

Data inspection

```
library(PerformanceAnalytics)
```

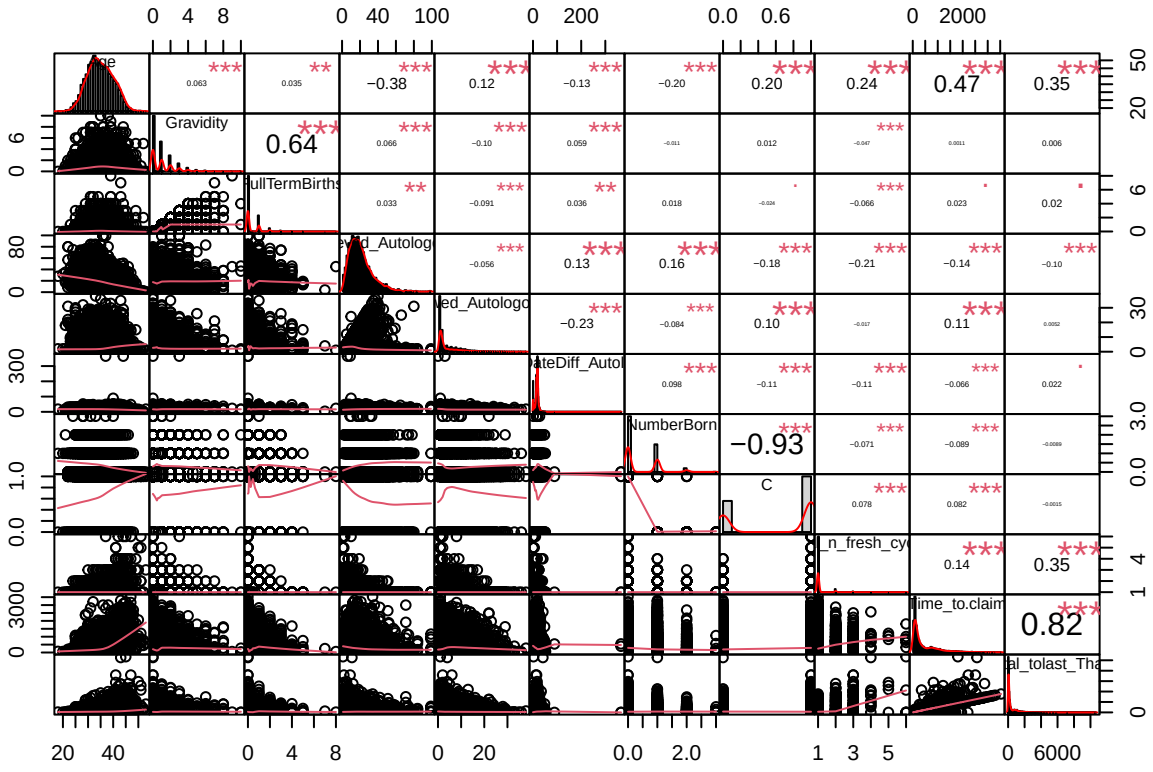
```
## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

##
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':
##
##     first, last

##
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':
##
##     legend
```

```
df.numeric <- df %>% dplyr::select(where(is.numeric))
chart.Correlation(df.numeric, histogram=TRUE, pch="+")
```

```
## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter
```

```
## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter
```

```
## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter
```



Model selection: I used logistic regressions as a bassline comparative fit to model Class as a function of my predictors. The first method I used was "all variable included" logistic regression. Then used a stepwise selection process to narrow down the predictors from the initial regression to interpret which factors are the most important to accurately classifying live birth. The predictive ML models were assessed on the basis of the objectives: Penelaized regressions (lasso ridge), Random Forest, and SVM for predicting IVF outcome of live birth.

I have splitted 75% of the data for training using our new factorized variable and the remaining 25% for testing.

Data Partition

```
## 75% of the sample size
smp_size <- floor(0.75 * nrow(df))

## set the seed
set.seed(2050)
train_ind <- sample(seq_len(nrow(df)), size = smp_size)

train <- df[train_ind, ]
test <- df[-train_ind, ]
```

Automated regression and logistic regression

stepwise variable selection process to narrow down the predictors in the regression to interpret which factors are the most important to accurately classifying the response variable.

The estimates from logistic regression characterize the relationship between the predictors and response variable on a log-odds scale.

```
#build models
logreg <- glm(PregnancyOutcome_Live.Birth ~ Clinic.Region.USA + Age +  DonorIdentityKnown +  Gravidity +
TubalOther + Uterine + Unexplained + OtherNonInfertile +  OtherPGD + Unknowen + elect + DOR..35 + Thawed

logreg_steps <- glm(PregnancyOutcome_Live.Birth ~ Clinic.Region.USA + Age +  DonorIdentityKnown +  Gravi
TubalOther + Uterine + Unexplained + OtherNonInfertile +  OtherPGD + Unknowen + elect + DOR..35 + Thawed
step.model <- stepAIC(logreg, trace = FALSE)

#prediction on test data
log.predict <- predict(logreg, newdata = test, type = 'response')
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type ==
step.predict <- predict(logreg_steps, newdata = test, type = 'response')
```

After training and testing the model, it's important to understand how well that model did in regards to its accuracy and predictive power. Our models accurately predicted 65% of the observations in the testing set.

Logistic Regression

```
#confusion matrix
log.prediction.rd <- ifelse(log.predict > 0.5, 1, 0)
step.prediction <- ifelse(step.predict > .5, 1, 0)

table_2 <- table(log.prediction.rd, test$PregnancyOutcome_Live.Birth)
table_3 <- table(step.prediction, test$PregnancyOutcome_Live.Birth)

Accuracy_logreg <- sum(diag(table_2))/(sum(table_2))
Accuracy_stepwise <- sum(diag(table_3))/(sum(table_3))

labels <- c("Logisitic Regression", "LogReg with Stepwise")
values <- c(Accuracy_logreg, Accuracy_stepwise)

Accuracy_table <- data.frame("Model" = labels, "Accuracy Rate" = values)
Accuracy_table
```

```
##                   Model Accuracy.Rate
## 1 Logisitic Regression     0.6525994
## 2 LogReg with Stepwise     0.6538226
```

Prediction & Confusion Matrix – test data

```
step.predict <- predict(logreg_steps, newdata = test, type = 'response')
```

A fitted model of the data in which to do predictions now we will check our fit against testing dataset.

summarizing and visualizing regression models

```
library(jtools)
```

```
##
## Attaching package: 'jtools'

## The following object is masked from 'package:arsenal':
##
##     %nin%
```

```
## The following object is masked from 'package:Hmisc':
##
##     %nin%
```

```
summ(step.model)
```

```
## Warning in !is.null(rmarkdown::metadata$output) && rmarkdown::metadata$output
## %in% : 'length(x) = 2 > 1' in coercion to 'logical(1)'
```

Observations	4903
Dependent variable	PregnancyOutcome_Live.Birth
Type	Generalized linear model
Family	binomial
Link	logit

$\chi^2(14)$	441.13
Pseudo-R^2 (Cragg-Uhler)	0.12
Pseudo-R^2 (McFadden)	0.07
AIC	5996.20
BIC	6093.66

	Est.	S.E.	z val.	p
(Intercept)	0.95	0.33	2.82	0.00
Clinic.Region.USANortheast	0.00	0.10	0.04	0.97
Clinic.Region.USASouth	0.13	0.09	1.38	0.17
Clinic.Region.USAWest	0.23	0.11	2.17	0.03
Age	-0.05	0.01	-6.49	0.00
DonorIdentityKnown1	0.21	0.14	1.56	0.12
Gravidity	-0.11	0.03	-3.47	0.00
FullTermBirths	0.16	0.06	2.73	0.01
OtherNonInfertile1	-0.28	0.20	-1.44	0.15
OtherPGD1	-0.44	0.22	-2.02	0.04
ThawedOocyte1	-0.89	0.08	-11.25	0.00
SpermSource_Donor1	0.42	0.14	2.93	0.00
NumberRetrieved_AutologousRetrieval1	0.01	0.00	3.90	0.00
ThawDateStartDateDiff_AutologousRetrieval1	0.00	0.00	1.50	0.13
Time_to.claim	0.00	0.00	1.64	0.10

Standard errors: MLE

Random Forest Data Partition

```
## 75% of the sample size
smp_size <- floor(0.75 * nrow(df))

## set the seed
```

```r
set.seed(2050)
train_ind <- sample(seq_len(nrow(df)), size = smp_size)

train.x <- df[train_ind, ]
test.x <- df[-train_ind, ]
```

```r
rf1 <- train(PregnancyOutcome_Live.Birth ~ Age + Gravidity +  FullTermBirths + Clinic.Region.USA +  Mal
                    data= train.x ,
                    method ="rf",
                    TuneLength = 5,
                    proximity=TRUE,
                    trControl = trainControl(method = "cv", number=12 ))
# OtherNonInfertile + ThawedEmbryo + TransferAttempted + DonorIdentityKnown  + OtherRFA + SpermSource_P
print(rf1)
```

```
## Random Forest
##
## 4903 samples
##   20 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (12 fold)
## Summary of sample sizes: 4495, 4495, 4494, 4494, 4495, 4495, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.6430759  0.01705351
##   12    0.6353329  0.14256025
##   22    0.6349149  0.14741632
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```r
rf1$results
```

```
##   mtry  Accuracy      Kappa  AccuracySD    KappaSD
## 1    2 0.6430759 0.01705351 0.003954829 0.01251731
## 2   12 0.6353329 0.14256025 0.013509730 0.02675366
## 3   22 0.6349149 0.14741632 0.014774287 0.03453317
```

```r
#p.rf <- predict(rf, train.x)
#confusionMatrix(p.rf, train.x$PregnancyOutcome_Live.Birth)
```

```r
library(randomForest)
```

```
## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
rf <- randomForest(PregnancyOutcome_Live.Birth ~Age + Gravidity +  FullTermBirths + Clinic.Region.USA +
                     data= train.x ,
                    TuneLength = 5,
                    importance = T)
# OtherNonInfertile + ThawedEmbryo + TransferAttempted + DonorIdentityKnown  + OtherRFA + SpermSource_P
```

Prediction & Confusion Matrix – train data:

```
p.rf <- predict(rf, train.x)
confusionMatrix(p.rf, train.x$PregnancyOutcome_Live.Birth)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3099  343
##          1   39 1422
##
##                Accuracy : 0.9221
##                  95% CI : (0.9142, 0.9294)
##     No Information Rate : 0.64
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8243
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9876
##             Specificity : 0.8057
##          Pos Pred Value : 0.9003
##          Neg Pred Value : 0.9733
##              Prevalence : 0.6400
##          Detection Rate : 0.6321
##    Detection Prevalence : 0.7020
##       Balanced Accuracy : 0.8966
##
##        'Positive' Class : 0
##
```

Prediction & Confusion Matrix – test data

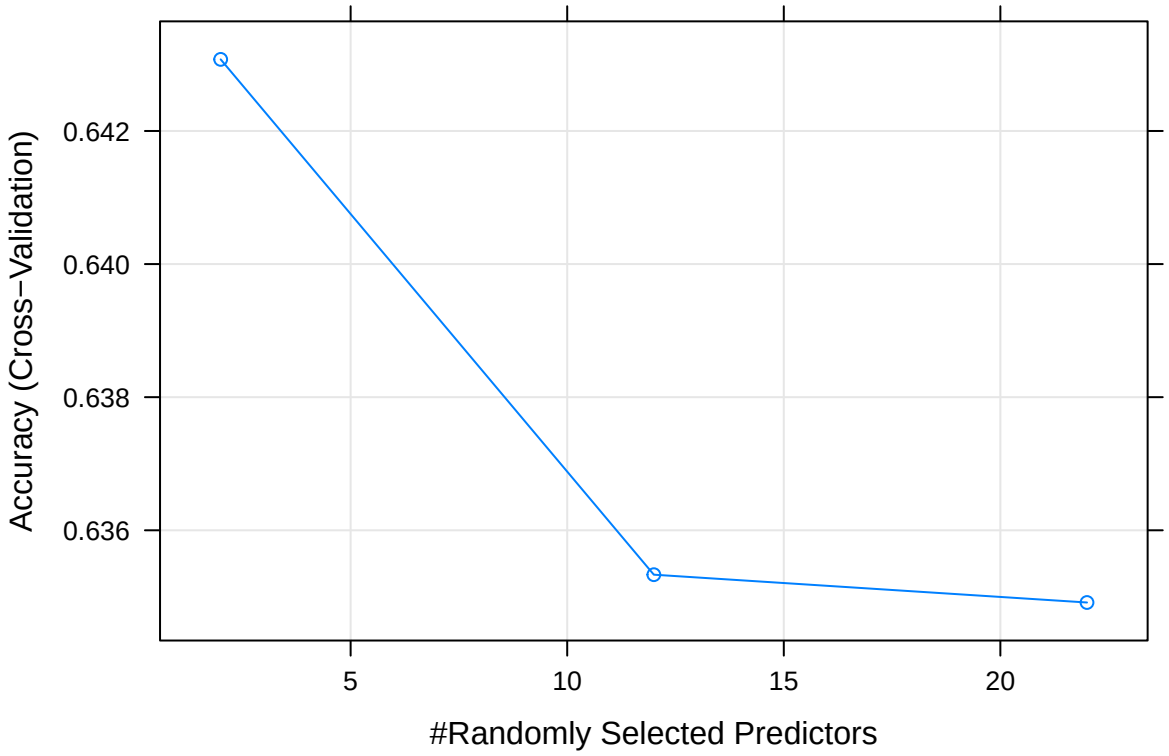
```
p2 <- predict(rf1, test.x)
confusionMatrix(p2, test.x$PregnancyOutcome_Live.Birth)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1048  572
##          1    6    9
##
##                Accuracy : 0.6465
##                  95% CI : (0.6228, 0.6697)
##     No Information Rate : 0.6446
##     P-Value [Acc > NIR] : 0.4496
##
##                   Kappa : 0.0125
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.99431
##             Specificity : 0.01549
##          Pos Pred Value : 0.64691
##          Neg Pred Value : 0.60000
##              Prevalence : 0.64465
##          Detection Rate : 0.64098
##    Detection Prevalence : 0.99083
##       Balanced Accuracy : 0.50490
##
##        'Positive' Class : 0
##
```

Variable Importance: These are the variables that the algorithm detected as the most important.

Error rate of Random Forest

```
plot(rf1)
```

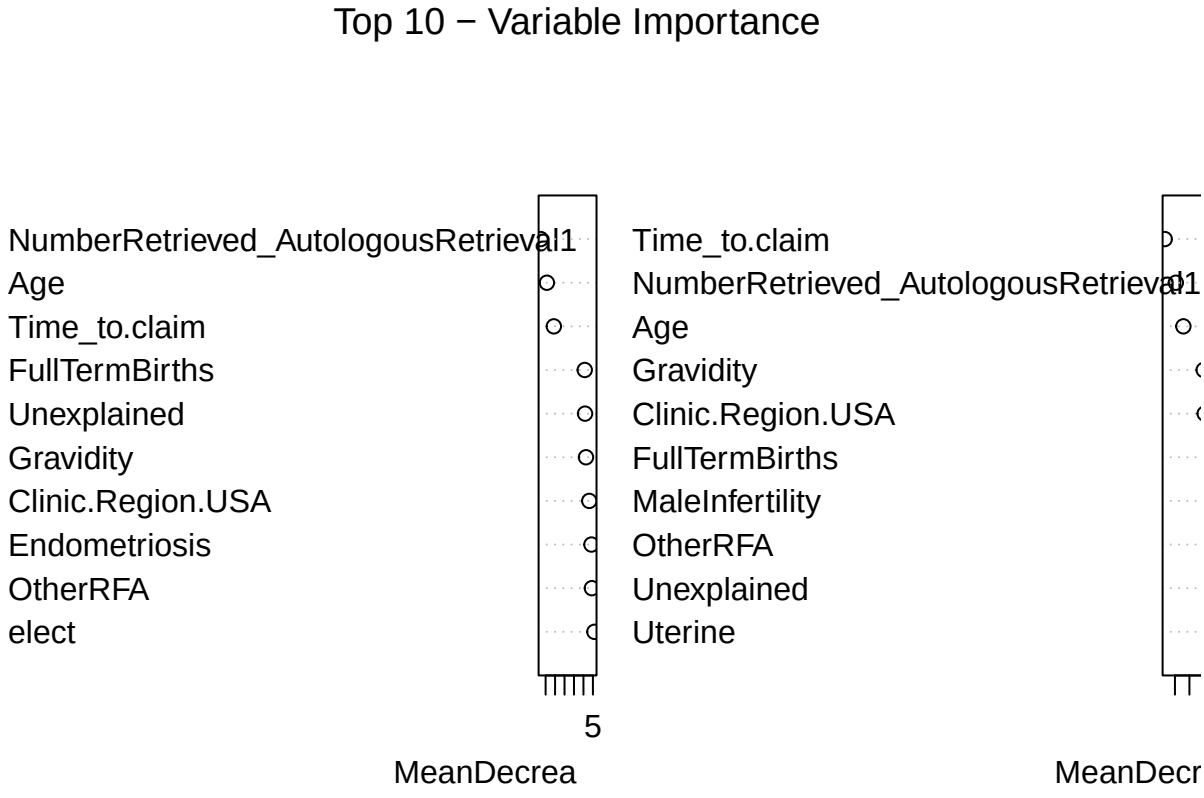



```
randomForest::importance(rf)
```

```
##                                                0          1 MeanDecreaseAccuracy
## Age                                     21.1373168 15.1804459          29.32000652
## Gravidity                               12.0829618 -3.9849106           8.66569697
## FullTermBirths                           6.9072198  4.2775862           9.32301352
## Clinic.Region.USA                       10.0656173 -1.1274839           6.92108101
## MaleInfertility                          6.8732963 -5.2651275           2.79947929
## Endometriosis                            8.6172719 -3.6001300           5.77307633
## Uterine                                  4.1349941 -3.1824587           1.62874717
## Unexplained                             10.7795868 -1.3952346           9.15834384
## OtherPGD                                -0.4318193  4.4912132           2.38385872
## Unknowen                                 1.8832040 -1.9439309           0.35464724
## elect                                    6.6169407 -3.6682153           4.22976961
## Onco                                    -2.0301903 -0.9688078          -2.35324920
## OtherRFA                                 7.1439843 -1.4462179           5.54593071
## Medical.benign                           8.7151827 -6.3772530           3.33475612
## Medical.Tumor.prevention                -1.4170276  0.0000000          -1.41700635
## Syndromatic.DOR                          0.8278800 -1.7064381          -0.35759835
## SpermSource_Donor                       -0.7819207  1.1817737           0.06338877
## NumberRetrieved_AutologousRetrieval1    29.1201943 12.3538728          32.56969102
## Time_to.claim                           18.5262063 13.7945995          25.65224684
## total_n_fresh_cycles                    -2.4963791 -1.6457848          -3.36411735
##                                        MeanDecreaseGini
## Age                                        241.93570782
```

```
## Gravidity                                101.37676403
## FullTermBirths                            57.67416894
## Clinic.Region.USA                         94.71820952
## MaleInfertility                           34.87168907
## Endometriosis                             14.95877996
## Uterine                                   21.42440856
## Unexplained                               25.33807327
## OtherPGD                                   8.58456114
## Unknowen                                  13.28501182
## elect                                     17.69449510
## Onco                                       3.47788063
## OtherRFA                                  26.07849389
## Medical.benign                            17.34196303
## Medical.Tumor.prevention                   0.07862201
## Syndromatic.DOR                            1.40021994
## SpermSource_Donor                         20.26120342
## NumberRetrieved_AutologousRetrieval1     293.93972278
## Time_to.claim                            370.57731563
## total_n_fresh_cycles                      20.93176148
```

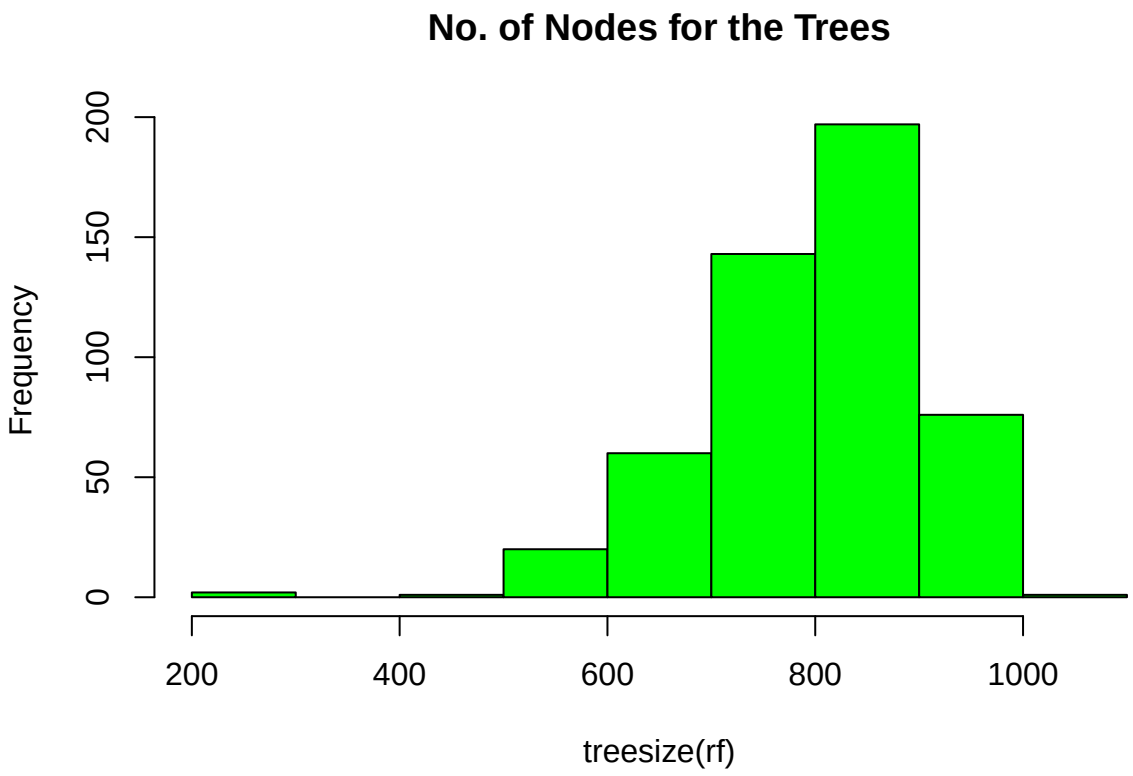
```
#Variable Importance
varImpPlot(rf,
           sort = T,
           n.var = 10,
           main = "Top 10 - Variable Importance")
```



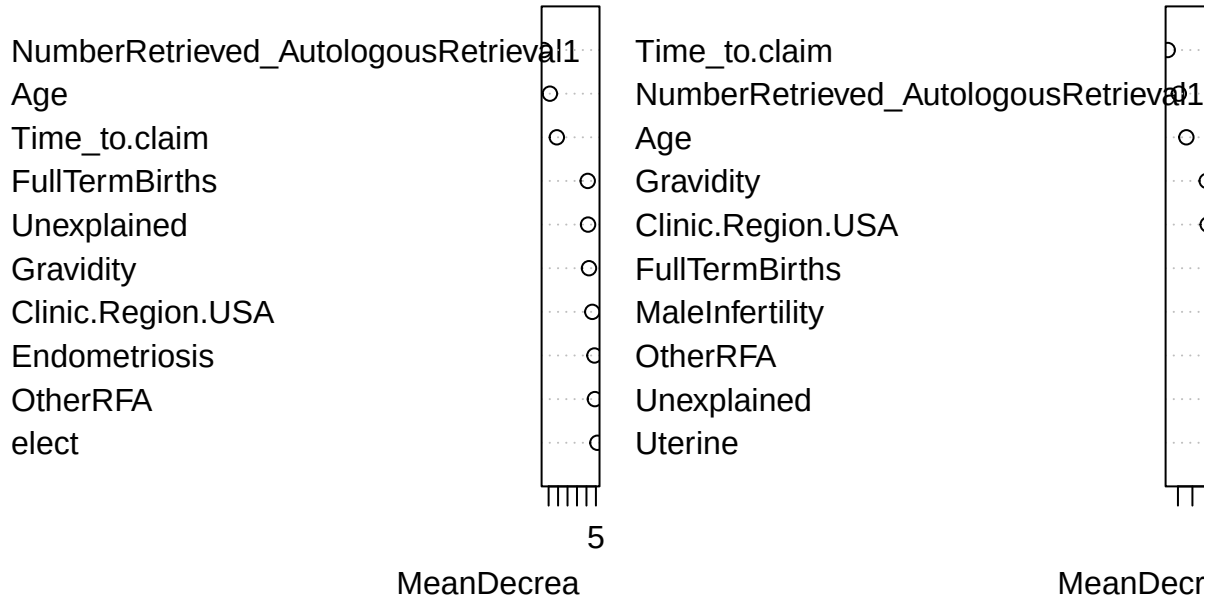
```
importance(rf)
```

```
##                                                0          1 MeanDecreaseAccuracy
## Age                                    21.1373168 15.1804459           29.32000652
## Gravidity                              12.0829618 -3.9849106            8.66569697
## FullTermBirths                          6.9072198  4.2775862            9.32301352
## Clinic.Region.USA                      10.0656173 -1.1274839            6.92108101
## MaleInfertility                         6.8732963 -5.2651275            2.79947929
## Endometriosis                           8.6172719 -3.6001300            5.77307633
## Uterine                                 4.1349941 -3.1824587            1.62874717
## Unexplained                            10.7795868 -1.3952346            9.15834384
## OtherPGD                               -0.4318193  4.4912132            2.38385872
## Unknowen                                1.8832040 -1.9439309            0.35464724
## elect                                   6.6169407 -3.6682153            4.22976961
## Onco                                   -2.0301903 -0.9688078           -2.35324920
## OtherRFA                                7.1439843 -1.4462179            5.54593071
## Medical.benign                          8.7151827 -6.3772530            3.33475612
## Medical.Tumor.prevention               -1.4170276  0.0000000           -1.41700635
## Syndromatic.DOR                         0.8278800 -1.7064381           -0.35759835
## SpermSource_Donor                      -0.7819207  1.1817737            0.06338877
## NumberRetrieved_AutologousRetrieval1   29.1201943 12.3538728           32.56969102
## Time_to.claim                          18.5262063 13.7945995           25.65224684
## total_n_fresh_cycles                   -2.4963791 -1.6457848           -3.36411735
##                                        MeanDecreaseGini
## Age                                        241.93570782
## Gravidity                                  101.37676403
## FullTermBirths                              57.67416894
## Clinic.Region.USA                           94.71820952
## MaleInfertility                             34.87168907
## Endometriosis                               14.95877996
## Uterine                                     21.42440856
## Unexplained                                 25.33807327
## OtherPGD                                     8.58456114
## Unknowen                                    13.28501182
## elect                                       17.69449510
## Onco                                         3.47788063
## OtherRFA                                    26.07849389
## Medical.benign                              17.34196303
## Medical.Tumor.prevention                     0.07862201
## Syndromatic.DOR                              1.40021994
## SpermSource_Donor                           20.26120342
## NumberRetrieved_AutologousRetrieval1       293.93972278
## Time_to.claim                              370.57731563
## total_n_fresh_cycles                        20.93176148
```

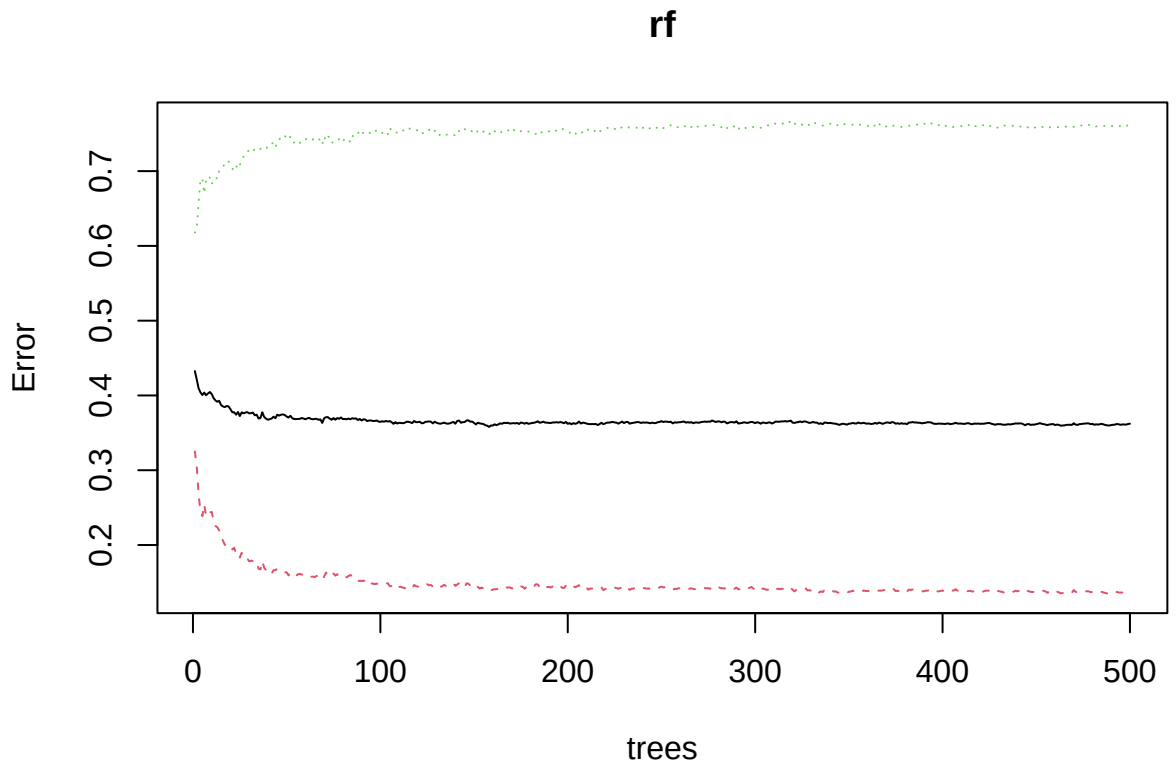
```
hist(treesize(rf),
     main = "No. of Nodes for the Trees",
     col = "green")
```




```
#Variable Importance
varImpPlot(rf,
           sort = T,
           n.var = 10,
           main = "Top 10 - Variable Importance")
```


Top 10 − Variable Importance
NumberRetrieved_AutologousRetrieval1
Age
Time_to.claim
FullTermBirths
Unexplained
Gravidity
Clinic.Region.USA
Endometriosis
OtherRFA
elect
5
MeanDecrea
Time_to.claim
NumberRetrieved_AutologousRetrieval1
Age
Gravidity
Clinic.Region.USA
FullTermBirths
MaleInfertility
OtherRFA
Unexplained
Uterine
MeanDecr

```
plot(rf)
```

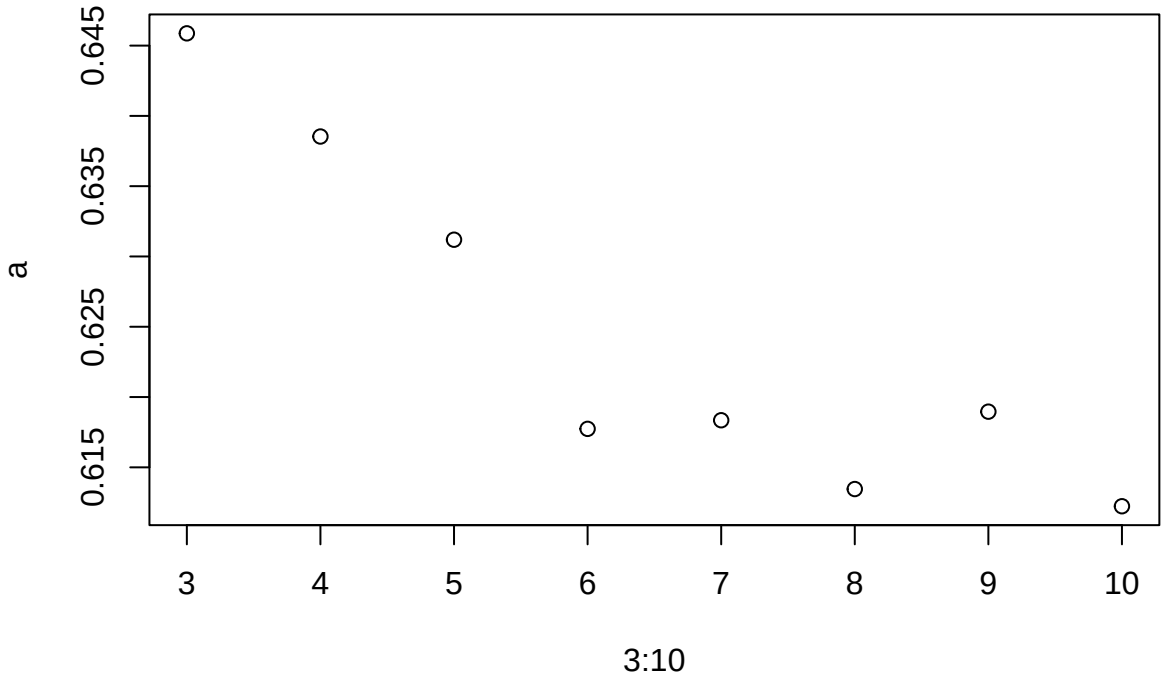


Tune my RF

```
a=c()
i=7
for (i in 2:10) {

rf.tune <- randomForest(PregnancyOutcome_Live.Birth ~ Age + Gravidity +  FullTermBirths + Clinic.Region
                    data= train.x ,
                    ntree= 400,
                    mtry=i,
                    importance= TRUE,
                    proximity=TRUE,
                    na.action=na.roughfix)

preValid <- predict(rf.tune, test.x, type = "class")
a[i-2] = mean(preValid == test.x$PregnancyOutcome_Live.Birth)
}
a
```

```
## [1] 0.6458716 0.6385321 0.6311927 0.6177370 0.6183486 0.6134557 0.6189602
## [8] 0.6122324
```

```
plot(3:10, a)
```



Final Model

```
rf.f <- randomForest(PregnancyOutcome_Live.Birth ~ Age + Gravidity +  FullTermBirths + Clinic.Region.US
                     data= train.x ,
                     ntree= 400,
                     mtry=2,
                     proximity=TRUE,
                     na.action=na.roughfix,
                     trControl = trainControl(method = "cv", number=23 ))
# OtherNonInfertile + ThawedEmbryo + TransferAttempted + DonorIdentityKnown  + OtherRFA + SpermSource_P
```

```
print(rf.f)
```

```
##
## Call:
##  randomForest(formula = PregnancyOutcome_Live.Birth ~ Age + Gravidity +      FullTermBirths + Clinic
##                Type of random forest: classification
##                      Number of trees: 400
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 35.55%
## Confusion matrix:
##      0  1 class.error
## 0 3106 32  0.01019758
## 1 1711 54  0.96940510
```

Prediction & Confusion Matrix – test data

```
p.final.t <- predict(rf.f, train.x)
confusionMatrix(p.final.t, train.x$PregnancyOutcome_Live.Birth)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3123 1667
##          1   15   98
##
##                Accuracy : 0.6569
##                  95% CI : (0.6435, 0.6702)
##     No Information Rate : 0.64
##     P-Value [Acc > NIR] : 0.00691
##
##                   Kappa : 0.0638
##
##  Mcnemar's Test P-Value : < 2e-16
##
##             Sensitivity : 0.99522
##             Specificity : 0.05552
##          Pos Pred Value : 0.65198
##          Neg Pred Value : 0.86726
##              Prevalence : 0.64002
##          Detection Rate : 0.63696
##    Detection Prevalence : 0.97695
##       Balanced Accuracy : 0.52537
##
##        'Positive' Class : 0
##
```

```
p.final <- predict(rf.f, test.x)
confusionMatrix(p.final, test.x$PregnancyOutcome_Live.Birth)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1046  564
##          1    8   17
##
##                Accuracy : 0.6502
##                  95% CI : (0.6265, 0.6733)
##     No Information Rate : 0.6446
##     P-Value [Acc > NIR] : 0.331
##
##                   Kappa : 0.0276
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.99241
```

```
##             Specificity : 0.02926
##          Pos Pred Value : 0.64969
##          Neg Pred Value : 0.68000
##              Prevalence : 0.64465
##          Detection Rate : 0.63976
##    Detection Prevalence : 0.98471
##       Balanced Accuracy : 0.51083
##
##        'Positive' Class : 0
##
```

```
treeT1 <- table(predict(rf.f, test.x, type = 'class'), test.x$PregnancyOutcome_Live.Birth)
Accuracyrf <- (treeT1[1,1]+treeT1[2,2])/(sum(treeT1))
Accuracyrf
```

```
## [1] 0.6501529
```

3 Lasso

```
# Install & load packages
if (!require(sas7bdat)) {install.packages("sas7bdat")}
if (!require(Hmisc)) {install.packages("Hmisc")}
if (!require(MASS)) {install.packages("MASS")}
if (!require(caret)) {install.packages("caret")}
if (!require(leaps)) {install.packages("leaps")}
if (!require(gamlr)) {install.packages("gamlr")}
if (!require(glmnet)) {install.packages("glmnet")}
```

Trim non factors before "hot-encoding"

```
#trim missing
trim <- c("ExternalPatientID ", "ExternalCycleId ")
df <- df[, !(names(df) %in% trim)]

df <- subset(df, select = -c(3,4, 35:38, 40:41 ,45, 46, 49))
```

Data Partition

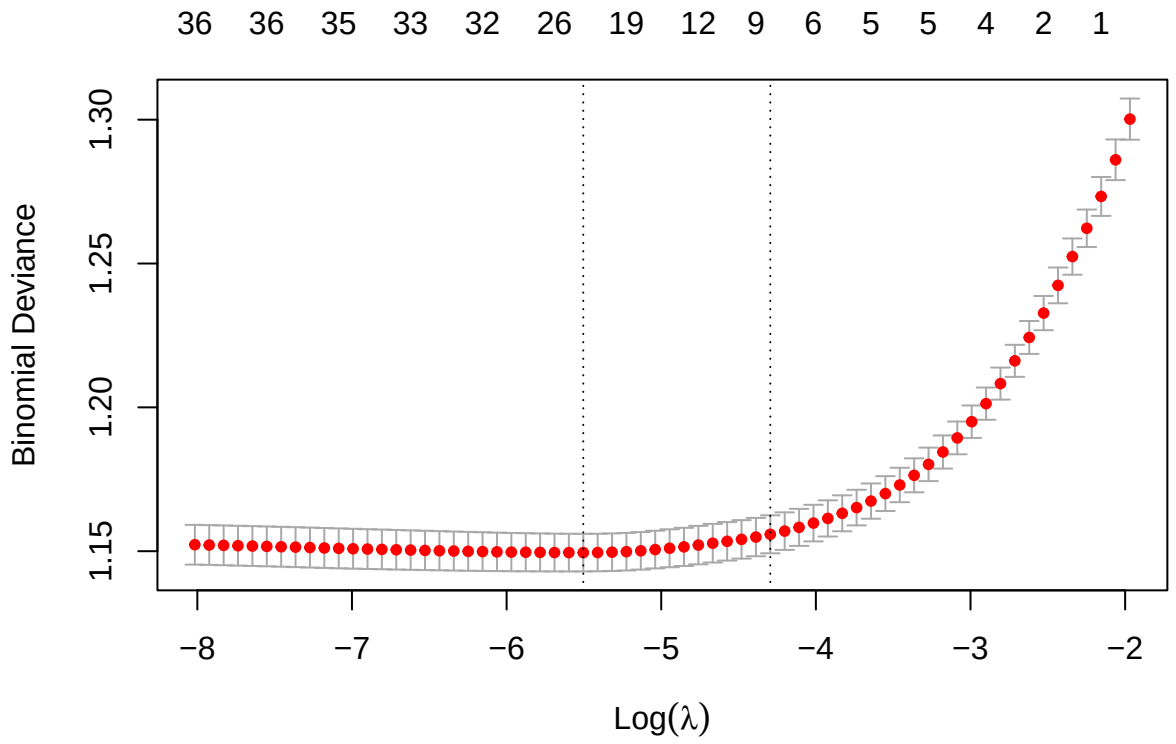
```
## 75% of the sample size
smp_size <- floor(0.75 * nrow(df))
## set the seed
set.seed(1732)
train_ind <- sample(seq_len(nrow(df)), size = smp_size)

train.lasso <- df[train_ind, ]
test.lasso <- df[-train_ind, ]
```

Create the matrix of predictors and also automatically converts categorical predictors to dummy variables

```
# Dumy code categorical predictor variables
x <- model.matrix(PregnancyOutcome_Live.Birth ~., train.lasso)[,-1]
y <- train.lasso$PregnancyOutcome_Live.Birth
```

Fit the lasso penalized regression model:

```
# Find the best lambda using cross-validation
#Find the optimal value of lambda that minimizes the cross-validation error
set.seed(123)
cv.lasso <- cv.glmnet(x, y, alpha = 1, family = "binomial")
# Fit the final model on the training data
model <- glmnet(x, y, alpha = 1, family = "binomial",
                lambda = cv.lasso$lambda.min)
plot(cv.lasso)
```




```
# Display regression coefficients
coef(model)
```

```
## 44 x 1 sparse Matrix of class "dgCMatrix"
##                                                s0
## (Intercept)                          -2.588045e+00
## Clinic.Region.USANortheast           -8.391288e-02
## Clinic.Region.USASouth                6.429167e-02
## Clinic.Region.USAWest                 1.353513e-01
```

```
## ReportingYear2015                          -5.391895e-02
## ReportingYear2016                           .
## ReportingYear2017                          -7.337966e-02
## ReportingYear2018                           1.232919e-02
## ReportingYear2019                           .
## Age                                        -3.589161e-02
## DonorIdentityKnown1                         .
## Gravidity                                  -3.912968e-02
## FullTermBirths                              5.504744e-02
## MaleInfertility1                            .
## Endometriosis1                              .
## PolycysticOvaries1                          .
## DiminishedOvarianReserve1                  -1.066338e-01
## TubalLigation1                              .
## TubalHydrosalpinx1                          .
## TubalOther1                                 .
## Uterine1                                    6.370508e-02
## Unexplained1                                .
## OtherRFA1                                  -1.277837e-01
## OtherNonInfertile1                         -8.826597e-03
## OtherPGD1                                   .
## Unknowen1                                   .
## elect1                                      .
## Onco1                                       .
## Medical.benign1                            -6.411613e-02
## Medical.Tumor.prevention1                   .
## Syndromatic.DOR1                            .
## DOR..351                                   -2.110633e-02
## FreshEmbryo1                               -3.025965e-01
## ThawedEmbryo1                               .
## ThawedOocyte1                              -4.464627e-04
## SpermSource_Partner1                       -2.431653e-01
## SpermSource_Donor1                          .
## SpermSource_Mixed1                          .
## TransferAttempted1                          3.446392e+00
## NumberRetrieved_AutologousRetrieval1        6.646750e-03
## NumberThawed_AutologousRetrieval1           2.142204e-02
## ThawDateStartDateDiff_AutologousRetrieval1  .
## total_n_fresh_cycles                       -1.546996e-02
## Time_to.claim                               1.202538e-05
```

```
# Make predictions on the test data
x.test <- model.matrix(PregnancyOutcome_Live.Birth ~., test.lasso)[,-1]
probabilities <- model %>% predict(newx = x.test)
predicted.classes <- ifelse(probabilities > 0.5, "1", "0")
# Model accuracy
observed.classes <- test.lasso$PregnancyOutcome_Live.Birth
mean(predicted.classes == observed.classes)
```

```
## [1] 0.6287462
```

```
cv.lasso$lambda.min
```

```
## [1] 0.004069824
```

```
cv.lasso$lambda.1se
```

```
## [1] 0.01364043
```

```
coef(cv.lasso, cv.lasso$lambda.min)
```

```
## 44 x 1 sparse Matrix of class "dgCMatrix"
##                                                       s1
## (Intercept)                                 -2.586293e+00
## Clinic.Region.USANortheast                  -8.408450e-02
## Clinic.Region.USASouth                       6.408261e-02
## Clinic.Region.USAWest                        1.351380e-01
## ReportingYear2015                           -5.397522e-02
## ReportingYear2016                            .
## ReportingYear2017                           -7.338561e-02
## ReportingYear2018                            1.235115e-02
## ReportingYear2019                            .
## Age                                         -3.588671e-02
## DonorIdentityKnown1                          .
## Gravidity                                   -3.930799e-02
## FullTermBirths                               5.523349e-02
## MaleInfertility1                             .
## Endometriosis1                               .
## PolycysticOvaries1                           .
## DiminishedOvarianReserve1                   -1.066847e-01
## TubalLigation1                               .
## TubalHydrosalpinx1                           .
## TubalOther1                                  .
## Uterine1                                     6.372669e-02
## Unexplained1                                 .
## OtherRFA1                                   -1.277893e-01
## OtherNonInfertile1                          -8.772398e-03
## OtherPGD1                                    .
## Unknowen1                                    .
## elect1                                       .
## Onco1                                        .
## Medical.benign1                             -6.409242e-02
## Medical.Tumor.prevention1                    .
## Syndromatic.DOR1                             .
## DOR..351                                    -2.107302e-02
## FreshEmbryo1                                -3.045325e-01
## ThawedEmbryo1                                .
## ThawedOocyte1                                .
## SpermSource_Partner1                        -2.424167e-01
## SpermSource_Donor1                           .
## SpermSource_Mixed1                           .
## TransferAttempted1                           3.444696e+00
## NumberRetrieved_AutologousRetrieval1         6.640387e-03
## NumberThawed_AutologousRetrieval1            2.147974e-02
## ThawDateStartDateDiff_AutologousRetrieval1   .
## total_n_fresh_cycles                        -1.526808e-02
## Time_to.claim                                1.213014e-05
```

Compute the final lasso modelusing lambda.min:

```
# Final model with lambda.min
lasso.model <- glmnet(x, y, alpha = 1, family = "binomial",
                      lambda = cv.lasso$lambda.min)
# Make prediction on test data
x.test <- model.matrix(PregnancyOutcome_Live.Birth ~., test.lasso)[,-1]
probabilities.lasso <- lasso.model %>% predict(newx = x.test)
predicted.classes.lasso <- ifelse(probabilities > 0.5, "1", "0")
# Model accuracy
observed.classes.lasso <- test.lasso$PregnancyOutcome_Live.Birth
Accuracy_lasso <- mean(predicted.classes.lasso == observed.classes.lasso)
```

Compute the full logistic model

```
# Fit the model
log.model <- glm(PregnancyOutcome_Live.Birth ~., data = train.lasso, family = binomial)
# Make predictions
probabilities.reg <- log.model %>% predict(test.lasso, type = "response")
predicted.classes.reg <- ifelse(probabilities > 0.5, "1", "0")
# Model accuracy
observed.classes.reg <- test.lasso$PregnancyOutcome_Live.Birth
mean(predicted.classes.reg == observed.classes.reg)
```

```
## [1] 0.6287462
```

4 SVM

Data Partition

```
## 75% of the sample size
smp_size <- floor(0.75 * nrow(df))
## set the seed
set.seed(1732)
train_ind <- sample(seq_len(nrow(df)), size = smp_size)

train <- df[train_ind, ]
test <- df[-train_ind, ]
```

Option 1 (as dataset not Matrix)

```
library(caTools)
library(e1071)
```

```
##
## Attaching package: 'e1071'

## The following objects are masked from 'package:PerformanceAnalytics':
##
##     kurtosis, skewness
```

```
## The following object is masked from 'package:Hmisc':
##
##     impute
```

```
#build model
tuning.para <- tune.svm(PregnancyOutcome_Live.Birth~.,
                        data = train, gamma = 10^-2,
                        cost = 10, tunecontrol = tune.control(cross=10))
svm.model <- tuning.para$best.model
```

```
#predict model
svm.pred <- predict(svm.model, test)
svm.pred <- as.data.frame(svm.pred)
```

```
svm.table <- table(svm.pred$svm.pred, test$PregnancyOutcome_Live.Birth)
Accuracy_svm <- sum(diag(svm.table))/(sum(svm.table))
Accuracy_svm
```

```
## [1] 0.6318043
```

Models Discrimination We wanted to assess the classifier performance. Using the proportion of positive data points that are correctly considered as positive and the proportion of negative data points that are mistakenly considered as positive. The area under the ROC curve is 0.660 and 0.662 indicate that the model is not very efficient in discriminating between live birth and no live birth outcome.

Model Discrimination ROC for logreg

```
test$p_e <- predict(step.model, type="response", newdata=test)
roccurve.stepwise <- roc(test$PregnancyOutcome_Live.Birth ~ as.numeric(test$p_e))
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
auc_stepwise <- roccurve.stepwise #
```

```
test$p_reg <- predict(log.model, type="response", newdata=test)
roccurve.reg <- roc(test$PregnancyOutcome_Live.Birth ~ as.numeric(test$p_reg))
```

```
## Setting levels: control = 0, case = 1
```

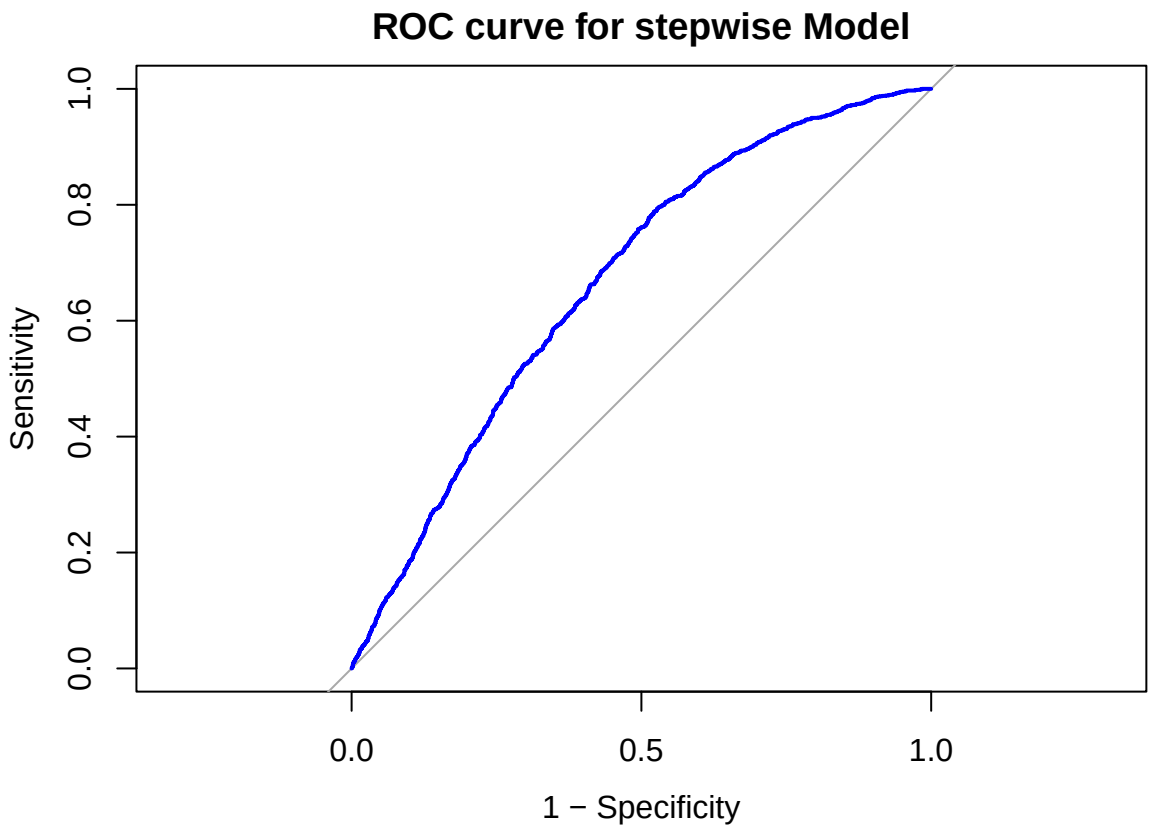
```
## Setting direction: controls < cases
```

```
auc_reg <- roccurve.reg #
```

```
train$p_a <- predict(step.model, type="response", newdata=train)
test$p_a <- predict(step.model, type="response", newdata=test)

roccurve.stepwise.tr <- roc(train$PregnancyOutcome_Live.Birth ~ train$p_a); roccurve.stepwise.tr #
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

##
## Call:
## roc.formula(formula = train$PregnancyOutcome_Live.Birth ~ train$p_a)
##
## Data: train$p_a in 3165 controls (train$PregnancyOutcome_Live.Birth 0) < 1738 cases (train$PregnancyO
## Area under the curve: 0.6681
```

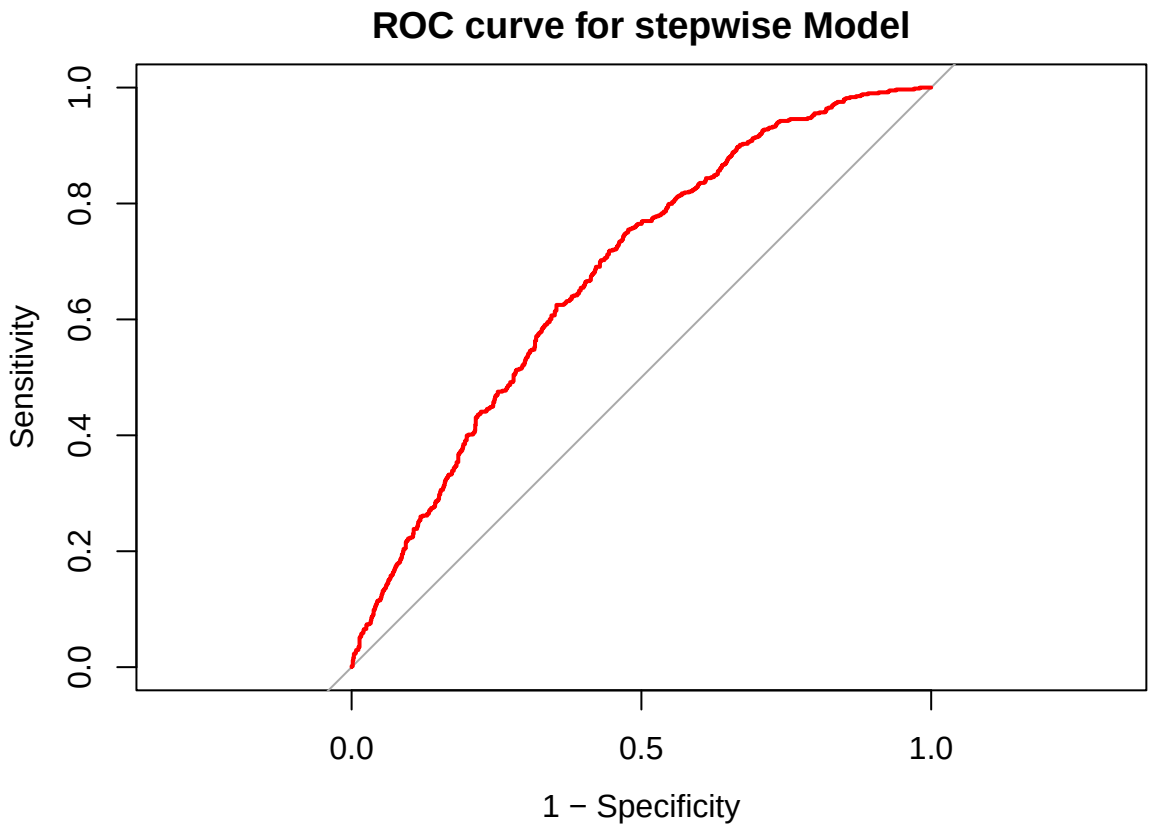
```
plot(roccurve.stepwise.tr, legacy.axes=T, main="ROC curve for stepwise Model", col="blue")
```




```
roccurve.stepwise.ts <- roc(test$PregnancyOutcome_Live.Birth ~ test$p_a); roccurve.stepwise.ts #
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

##
## Call:
## roc.formula(formula = test$PregnancyOutcome_Live.Birth ~ test$p_a)
##
## Data: test$p_a in 1027 controls (test$PregnancyOutcome_Live.Birth 0) < 608 cases (test$PregnancyOutco
## Area under the curve: 0.6788
```

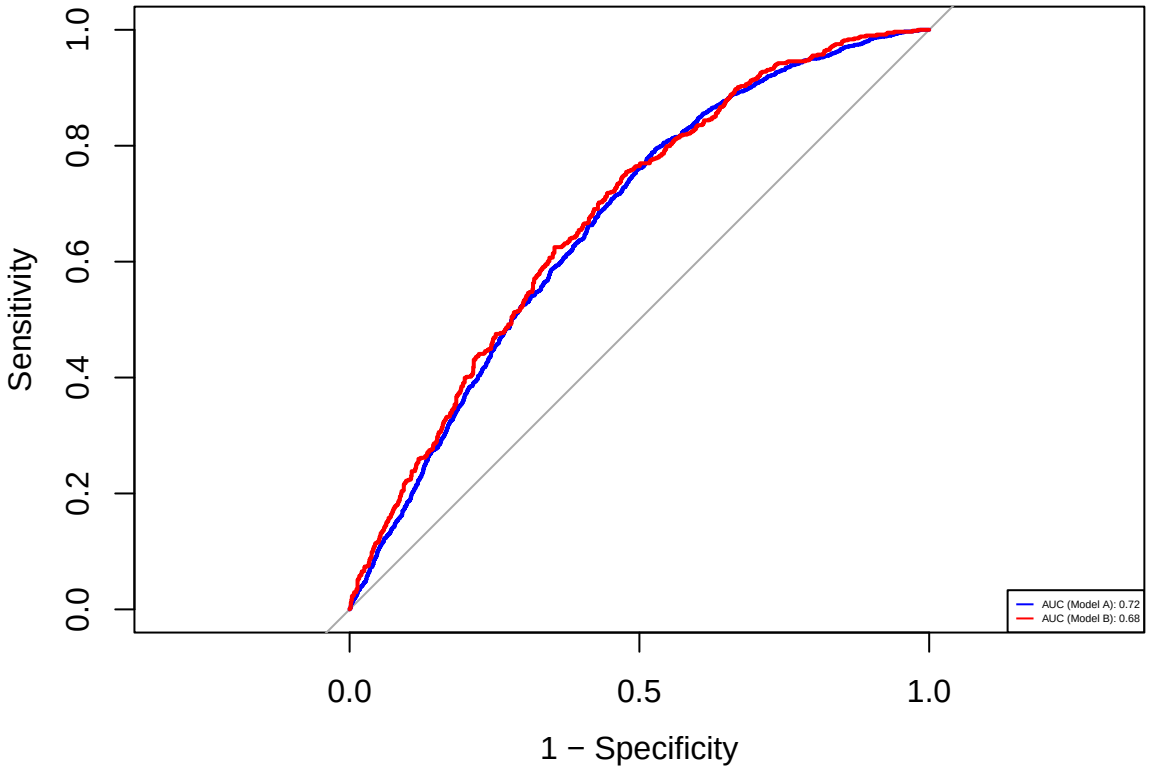
```
plot(roccurve.stepwise.ts, legacy.axes=T, main="ROC curve for stepwise Model", col="red")
```




```
# Compare AUCs using DeLong's test
roc.test(roccurve.stepwise.tr, roccurve.stepwise.ts, alternative="two.sided")
```

```
##
##  DeLong's test for two ROC curves
##
## data:  roccurve.stepwise.tr and roccurve.stepwise.ts
## D = -0.69397, df = 2841.5, p-value = 0.4878
## alternative hypothesis: true difference in AUC is not equal to 0
## sample estimates:
## AUC of roc1 AUC of roc2
##   0.6681230   0.6787879
```

```
# ROC curves of the two models
plot(roccurve.stepwise.tr, legacy.axes=T, col="blue"); plot(roccurve.stepwise.ts, legacy.axes=T, col="r
legend("bottomright", legend=c("AUC (Model A): 0.72", "AUC (Model B): 0.68"), col=c("blue", "red"), lty=
```



Discrimination for random forest

```
rf_prediction <- predict(rf, test.x, type ="prob")
# build the logistic regression model and test it
lr_model <- glm(PregnancyOutcome_Live.Birth ~ Age + Gravidity +  FullTermBirths + Clinic.Region.USA +  I
lr_prediction <- predict(lr_model, test.x, type = "response")

# ROC curves
ROC_rf <- roc(test.x$PregnancyOutcome_Live.Birth, rf_prediction[,2])
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

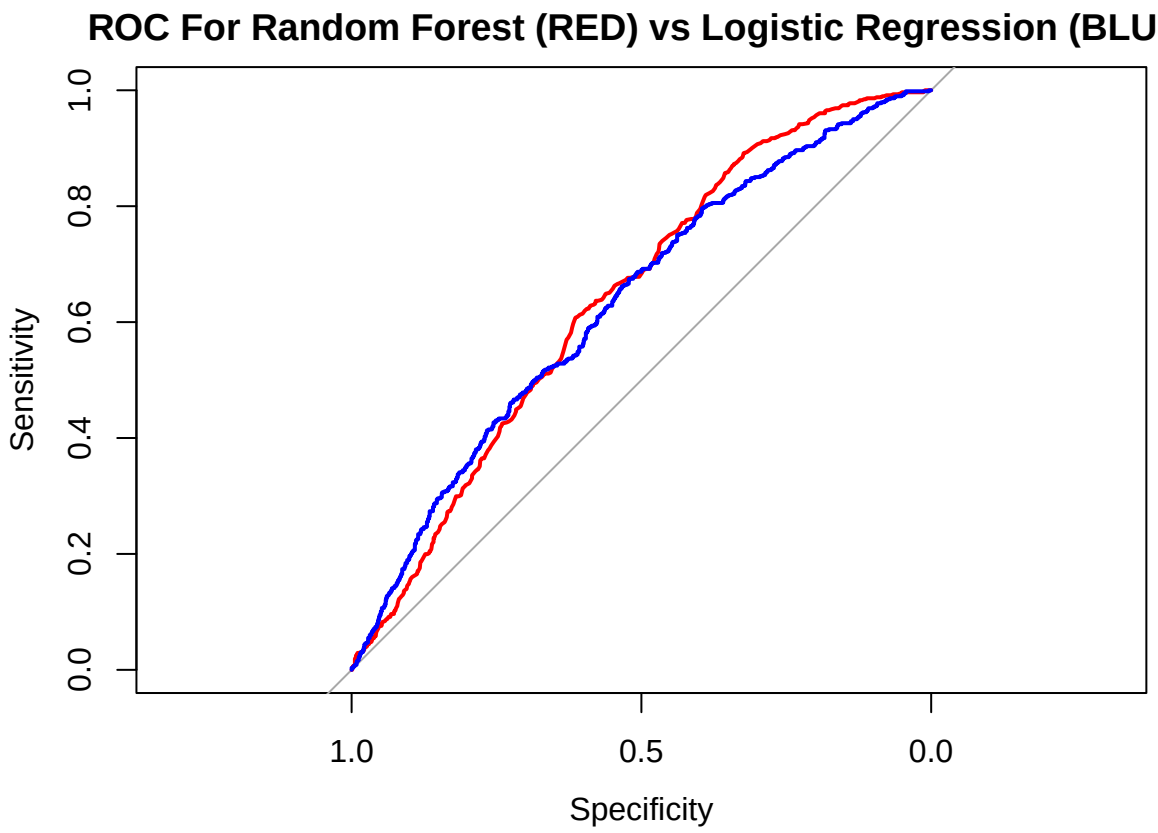
```
ROC_lr <- roc(test.x$PregnancyOutcome_Live.Birth, lr_prediction)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```
# Area Under Curve (AUC) for each ROC curve (higher -> better)
ROC_rf_auc <- auc(ROC_rf)
ROC_lr_auc <- auc(ROC_lr)
```

Discrimination

```
plot(ROC_rf, col = "red", main = "ROC For Random Forest (RED) vs Logistic Regression (BLUE)")
lines(ROC_lr, col = "blue")
```




```
# print the performance of each model
paste("Accuracy % of random forest: ", mean(test.x$PregnancyOutcome_Live.Birth == round(rf_prediction[,
```

```
## [1] "Accuracy % of random forest:  0.63302752293578"
```

```
paste("Accuracy % of logistic regression: ", mean(test.x$PregnancyOutcome_Live.Birth == round(lr_predic
```

```
## [1] "Accuracy % of logistic regression:  0.64954128440367"
```

```
paste("Area under curve of random forest: ", ROC_rf_auc)
```

```
## [1] "Area under curve of random forest:  0.64000512758543"
```

```
paste("Area under curve of logistic regression: ", ROC_lr_auc)
```

```
## [1] "Area under curve of logistic regression:  0.632477864834235"
```

Discrimination for Lasso

A list of cross-validated ROC data, one for each model along the path.The first line identifies the lambda value giving the best area under the curve (AUC). Then we plot all the ROC curves in grey and the "winner" in red.

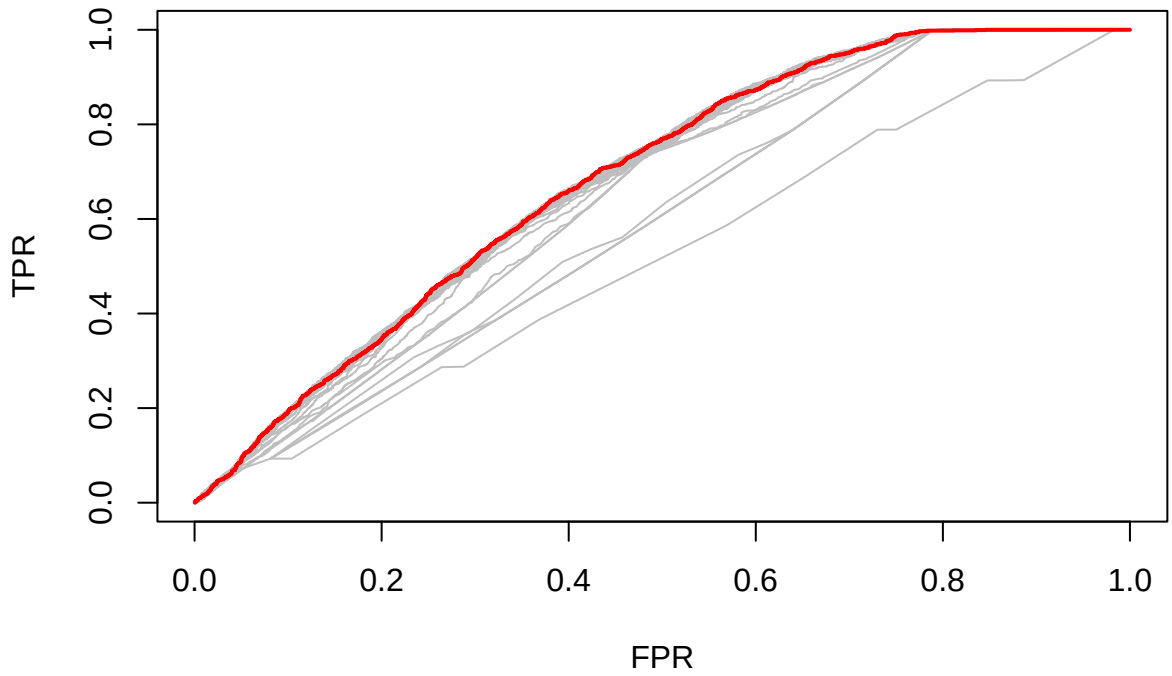
```
auc_lasso <- auc(test.lasso$PregnancyOutcome_Live.Birth, probabilities.lasso)
```

```
## Setting levels: control = 0, case = 1
```

```
## Warning in roc.default(response, predictor, auc = TRUE, ...): Deprecated use a
## matrix as predictor. Unexpected results may be produced, please pass a numeric
## vector.
```

```
## Setting direction: controls < cases
```

```
cfit <- cv.glmnet(x, y, family = "binomial", type.measure = "auc",
                  keep = TRUE)
rocs <- roc.glmnet(cfit$fit.preval, newy = y)

best <- cv.lasso$index["min",]
plot(rocs[[best]], type = "l")
invisible(sapply(rocs, lines, col="grey"))
lines(rocs[[best]], lwd = 2,col = "red")
```



Discrimination for SVM

```
test$p_e <- predict(svm.model, type="response", newdata=test)
roccurve.st.ts <- roc(test$PregnancyOutcome_Live.Birth ~ as.numeric(test$p_e))
```

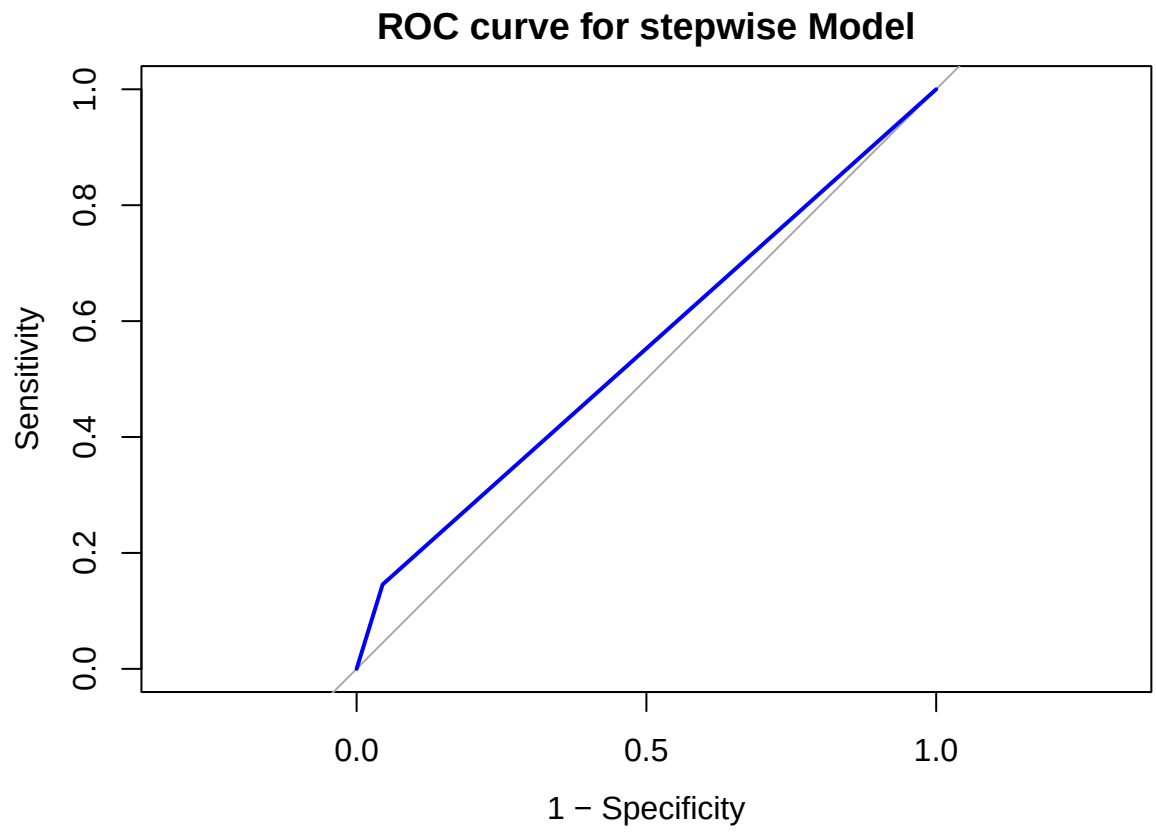
```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```

```
auc_svm <- roccurve.st.ts #
```

```
train$p_e <- predict(svm.model, type="response", newdata=train)
test$p_e <- predict(svm.model, type="response", newdata=test)

roccurve.st.tr <- roc(train$PregnancyOutcome_Live.Birth ~ as.numeric(train$p_e)); roccurve.st.tr #
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

##
## Call:
## roc.formula(formula = train$PregnancyOutcome_Live.Birth ~ as.numeric(train$p_e))
##
## Data: as.numeric(train$p_e) in 3165 controls (train$PregnancyOutcome_Live.Birth 0) < 1738 cases (tra
## Area under the curve: 0.5504
```

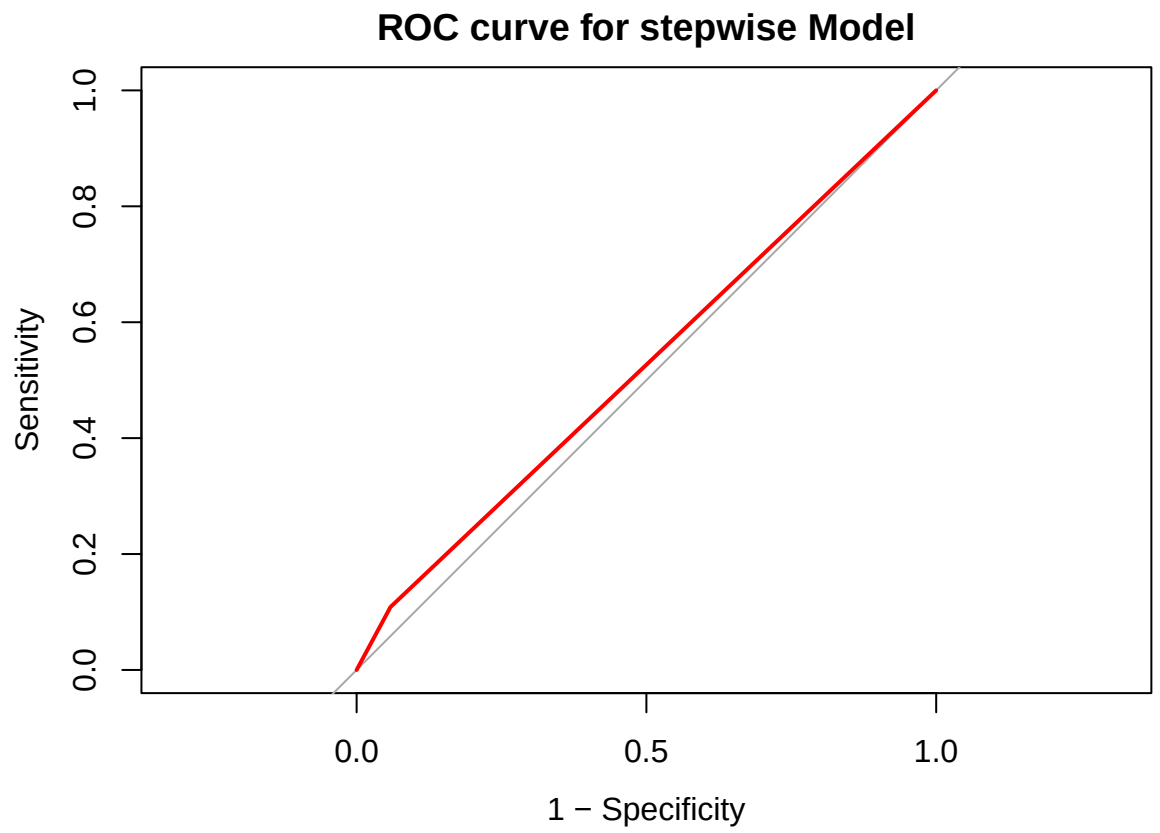
```
plot(roccurve.st.tr, legacy.axes=T, main="ROC curve for stepwise Model", col="blue")
```



```
roccurve.st.ts <- roc(test$PregnancyOutcome_Live.Birth ~ as.numeric(test$p_e)); roccurve.st.ts #
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

##
## Call:
## roc.formula(formula = test$PregnancyOutcome_Live.Birth ~ as.numeric(test$p_e))
##
## Data: as.numeric(test$p_e) in 1027 controls (test$PregnancyOutcome_Live.Birth 0) < 608 cases (test$P
## Area under the curve: 0.5251
```

```
plot(roccurve.st.ts, legacy.axes=T, main="ROC curve for stepwise Model", col="red")
```

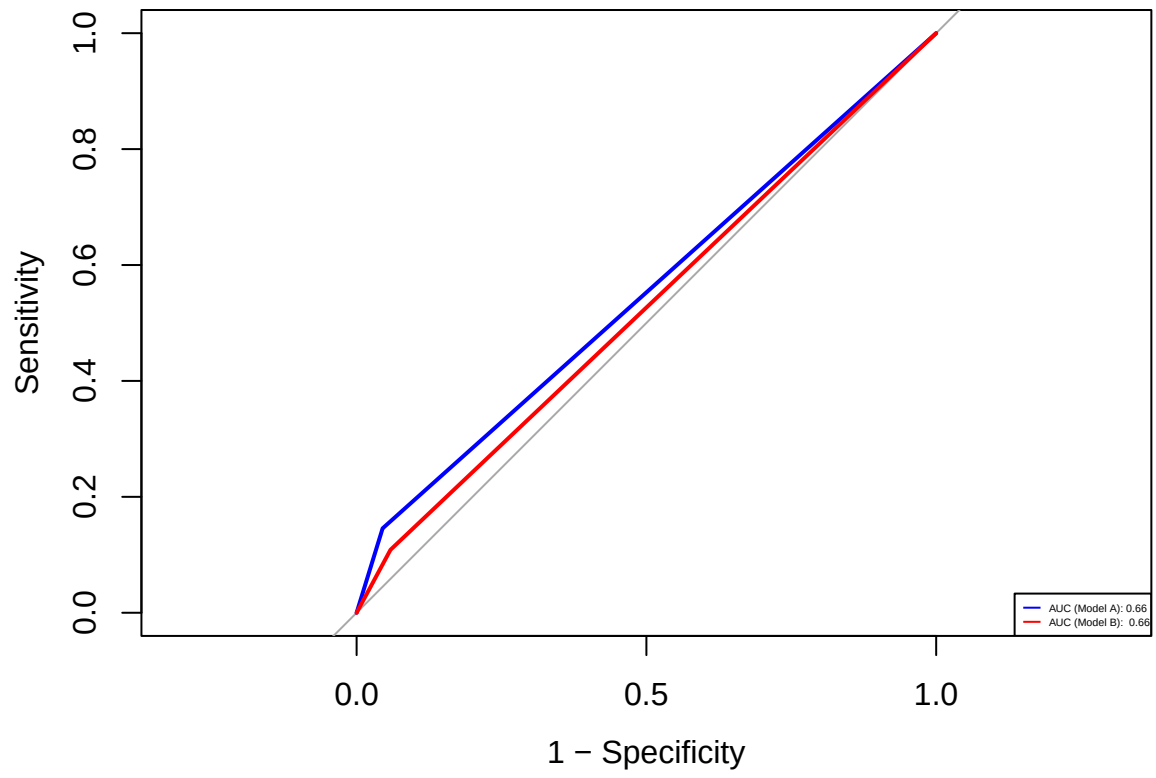



```
# Compare AUCs using DeLong's test
roc.test(roccurve.st.tr, roccurve.st.ts, alternative="two.sided")
```

```
##
##  DeLong's test for two ROC curves
##
## data:  roccurve.st.tr and roccurve.st.ts
## D = 2.9287, df = 3039.4, p-value = 0.003429
## alternative hypothesis: true difference in AUC is not equal to 0
## sample estimates:
```

```
## AUC of roc1 AUC of roc2
##    0.550352    0.525065
```

```
# ROC curves of the two models
plot(roccurve.st.tr, legacy.axes=T, col="blue"); plot(roccurve.st.ts, legacy.axes=T, col="red", add=T)
legend("bottomright", legend=c("AUC (Model A): 0.66", "AUC (Model B):  0.66"), col=c("blue", "red"), lty
```




```
AUC_logreg <- auc_reg$auc[1]
AUC_stepwise <-auc_stepwise$auc[1]
AUC_lasso <- auc_lasso[1]
AUC_rf <- auc(ROC_rf)[1]
AUC_svm <- auc_svm$auc[1]
```

For the project construct a comerative table with all the accuracy

```
labels.accuracy <- c("Logisitic Regression", "LogReg with Stepwise", "lasso", "Random Forest", "SVM")
values.accuracy <- c(Accuracy_logreg, Accuracy_stepwise, Accuracy_lasso, Accuracyrf, Accuracy_svm)
accuracytable <- data.frame("Model" = labels.accuracy, "Accuracy Rate" = values.accuracy)
accuracytable
```

```
##                 Model Accuracy.Rate
## 1 Logisitic Regression     0.6525994
## 2 LogReg with Stepwise     0.6538226
## 3                lasso     0.6287462
## 4        Random Forest     0.6501529
## 5                  SVM     0.6318043
```

```
labels.auc <- c("Logisitic Regression", "LogReg with Stepwise", "lasso", "Random Forest", "SVM")
values.auc <- c(AUC_logreg, AUC_stepwise, AUC_lasso, AUC_rf, AUC_svm)
auctable <- data.frame("Model" = labels.accuracy, "AUC" = values.auc )
auctable <- as.data.frame(auctable)
auctable
```

```
##                  Model       AUC
## 1 Logisitic Regression 0.6881262
## 2 LogReg with Stepwise 0.6787879
## 3                lasso 0.6940990
## 4        Random Forest 0.6400051
## 5                  SVM 0.5250650
```

```
SUM <- merge(accuracytable, auctable, by= "Model")
SUM
```

```
##                  Model Accuracy.Rate       AUC
## 1                lasso     0.6287462 0.6940990
## 2 Logisitic Regression     0.6525994 0.6881262
## 3 LogReg with Stepwise     0.6538226 0.6787879
## 4        Random Forest     0.6501529 0.6400051
## 5                  SVM     0.6318043 0.5250650
```

Conclusions: Based on the classification error rate, the stepwise logistic regression has the highest classification accuracy rate of 0.65. However, apart of SVM all of the models have very similar performance metrics so any of them would be appropriate to use.

Using ML modelson our predictive ability did not improved compered to baseic regression model . Many factors could account for this low performance including: quality of the data, the types of the modeling technique, and noisy data. We cannot account for other unknown omitted variables. Therefore, one should consider all of these factors when looking at the classification rate and determining whether it's 'good enough. I should consider revising the individual predictors that are in the model and consider if any other explanatory variables should be included.