Stroke Prediction Using Demographic and Clinical Variables

# Introduction

## Data description

The dataset is used to predict whether a patient is likely to get stroke based on the input parameters like gender, age, various diseases, and smoking status. Each row in the data provides relevant information about the patient and was posted on <https://www.kaggle.com/fedesoriano/stroke-prediction-dataset> by *Federico Soriano Palacios*.

Attribute Information:  
1) id: unique identifier  
2) gender: “Male”, “Female” or “Other”  
3) age: age of the patient  
4) hypertension: 0 if the patient doesn’t have hypertension, 1 if the patient has hypertension  
5) heart\_disease: 0 if the patient doesn’t have any heart diseases, 1 if the patient has a heart disease  
6) ever\_married: “No” or “Yes”  
7) work\_type: “children”, “Govt\_jov”, “Never\_worked”, “Private” or “Self-employed”  
8) Residence\_type: “Rural” or “Urban”  
9) avg\_glucose\_level: average glucose level in blood  
10) bmi: body mass index  
11) smoking\_status: “formerly smoked”, “never smoked”, “smokes” or “Unknown”\*  
12) stroke: 1 if the patient had a stroke or 0 if not  
\*Note: “Unknown” in smoking\_status means that the information is unavailable for this patient

library(tidyverse)  
library(caret)  
library(summarytools)  
library(corrplot)  
library(knitr)  
library(patchwork)  
library(mgcv)  
library(pROC)  
library(vip)  
library(ranger)  
  
opts\_chunk$set(  
 fig.width = 12,   
 fig.asp = .8,  
 out.width = "100%"  
)  
  
theme\_set(theme\_minimal() + theme(legend.position = "none"))  
options(  
 ggplot2.continuous.colour = "viridis",  
 ggplot2.continuous.fill = "viridis"  
)  
scale\_colour\_discrete = scale\_colour\_viridis\_d  
scale\_fill\_discrete = scale\_fill\_viridis\_d

# Load and clean the data  
stroke\_df =   
 read\_csv("./healthcare-dataset-stroke-data.csv") %>%   
 janitor::clean\_names() %>%   
 select(-id) %>%   
 mutate(  
 across(  
 c("gender", "hypertension", "heart\_disease", "ever\_married", "work\_type", "residence\_type", "smoking\_status", "stroke"),  
 ~ as.factor(.x)  
 )  
 ) %>%   
 mutate(  
 smoking\_status = na\_if(smoking\_status, "Unknown"),  
 bmi = na\_if(bmi, "N/A"),  
 bmi = as.numeric(bmi)  
 )  
  
# Summary of `stroke\_df`  
st\_options(  
 plain.ascii = FALSE,   
 style = "rmarkdown",   
 dfSummary.silent = TRUE,   
 footnote = NA,   
 subtitle.emphasis = FALSE  
)   
  
dfSummary(stroke\_df)

### Data Frame Summary

**stroke\_df**  
**Dimensions:** 5110 x 11  
**Duplicates:** 0

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | Variable | Stats / Values | Freqs (% of Valid) | Graph | Valid | Missing |
| 1 | gender [factor] | 1. Female 2. Male 3. Other | 2994 (58.6%) 2115 (41.4%) 1 ( 0.0%) | IIIIIIIIIII IIIIIIII | 5110 (100.0%) | 0 (0.0%) |
| 2 | age [numeric] | Mean (sd) : 43.2 (22.6) min < med < max: 0.1 < 45 < 82 IQR (CV) : 36 (0.5) | 104 distinct values | .   :       . : : : :   : : . : : : : : : : : : : : : : : : : : : : : : : : : : : : : | 5110 (100.0%) | 0 (0.0%) |
| 3 | hypertension [factor] | 1. 0 2. 1 | 4612 (90.3%) 498 ( 9.7%) | IIIIIIIIIIIIIIIIII I | 5110 (100.0%) | 0 (0.0%) |
| 4 | heart\_disease [factor] | 1. 0 2. 1 | 4834 (94.6%) 276 ( 5.4%) | IIIIIIIIIIIIIIIIII I | 5110 (100.0%) | 0 (0.0%) |
| 5 | ever\_married [factor] | 1. No 2. Yes | 1757 (34.4%) 3353 (65.6%) | IIIIII IIIIIIIIIIIII | 5110 (100.0%) | 0 (0.0%) |
| 6 | work\_type [factor] | 1. children 2. Govt\_job 3. Never\_worked 4. Private 5. Self-employed | 687 (13.4%) 657 (12.9%) 22 ( 0.4%) 2925 (57.2%) 819 (16.0%) | II II  IIIIIIIIIII III | 5110 (100.0%) | 0 (0.0%) |
| 7 | residence\_type [factor] | 1. Rural 2. Urban | 2514 (49.2%) 2596 (50.8%) | IIIIIIIII IIIIIIIIII | 5110 (100.0%) | 0 (0.0%) |
| 8 | avg\_glucose\_level [numeric] | Mean (sd) : 106.1 (45.3) min < med < max: 55.1 < 91.9 < 271.7 IQR (CV) : 36.8 (0.4) | 3979 distinct values | : . : : : . : : : : : : : . . . . . | 5110 (100.0%) | 0 (0.0%) |
| 9 | bmi [numeric] | Mean (sd) : 28.9 (7.9) min < med < max: 10.3 < 28.1 < 97.6 IQR (CV) : 9.6 (0.3) | 418 distinct values | : .   : :   : :   : : . : : : : . | 4909 (96.1%) | 201 (3.9%) |
| 10 | smoking\_status [factor] | 1. formerly smoked 2. never smoked 3. smokes 4. Unknown | 885 (24.8%) 1892 (53.1%) 789 (22.1%) 0 ( 0.0%) | IIII IIIIIIIIII IIII | 3566 (69.8%) | 1544 (30.2%) |
| 11 | stroke [factor] | 1. 0 2. 1 | 4861 (95.1%) 249 ( 4.9%) | IIIIIIIIIIIIIIIIIII | 5110 (100.0%) | 0 (0.0%) |

# Only 1 observation in the "Other" category of `gender`.  
# 1544 (30.2%) missing values in `smoking\_status` and 201 (3.9%) missing values in `bmi`.  
  
# Further clean the data  
stroke\_clean =   
 stroke\_df %>%   
 select(-smoking\_status) %>%   
 filter(gender != "Other") %>%   
 mutate(gender = droplevels(gender)) %>%   
 mutate(  
 across(c(hypertension, heart\_disease, stroke), ~ as.factor(recode(.x, "1" = "Yes", "0" = "No"))),  
 stroke = relevel(stroke, ref = "No")  
 )  
  
# Summary of `stroke\_clean`  
dfSummary(stroke\_clean)

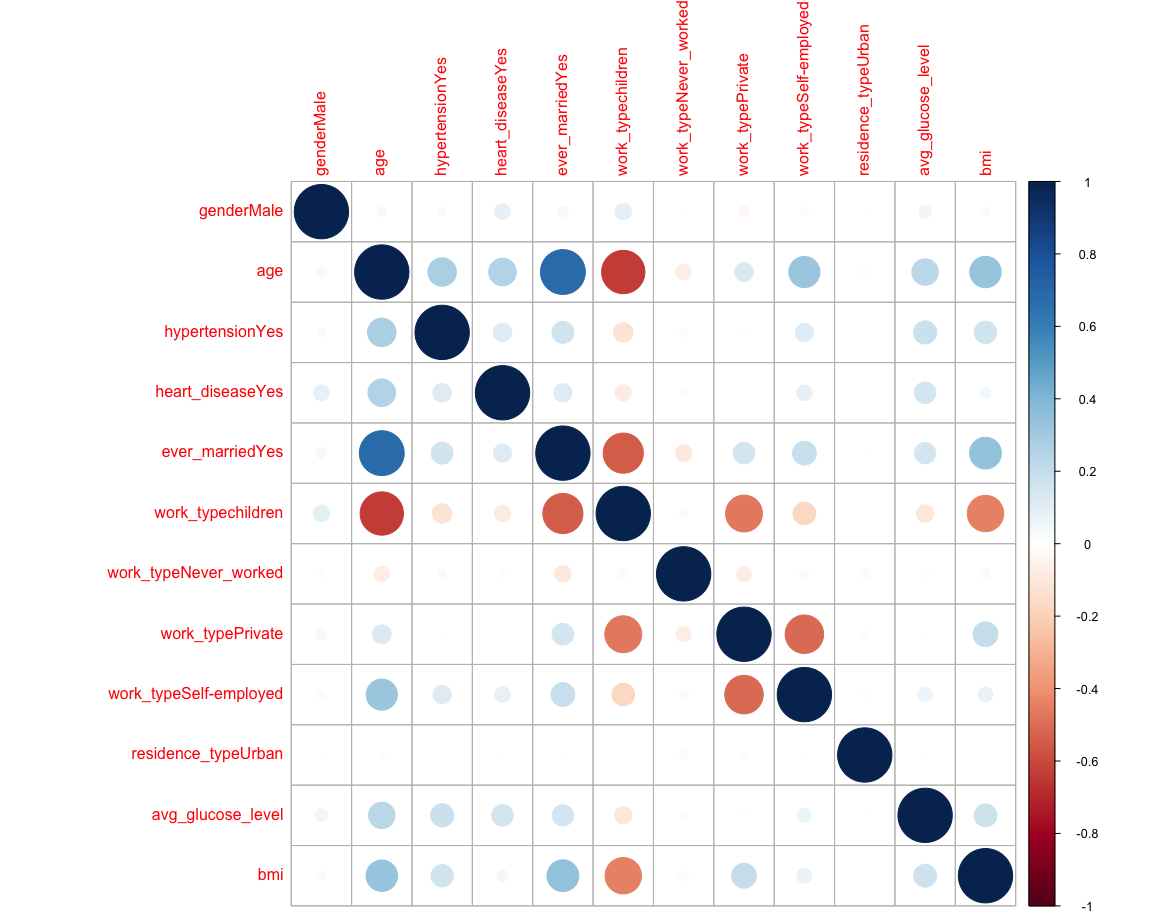
### Data Frame Summary

**stroke\_clean**  
**Dimensions:** 5109 x 10  
**Duplicates:** 0

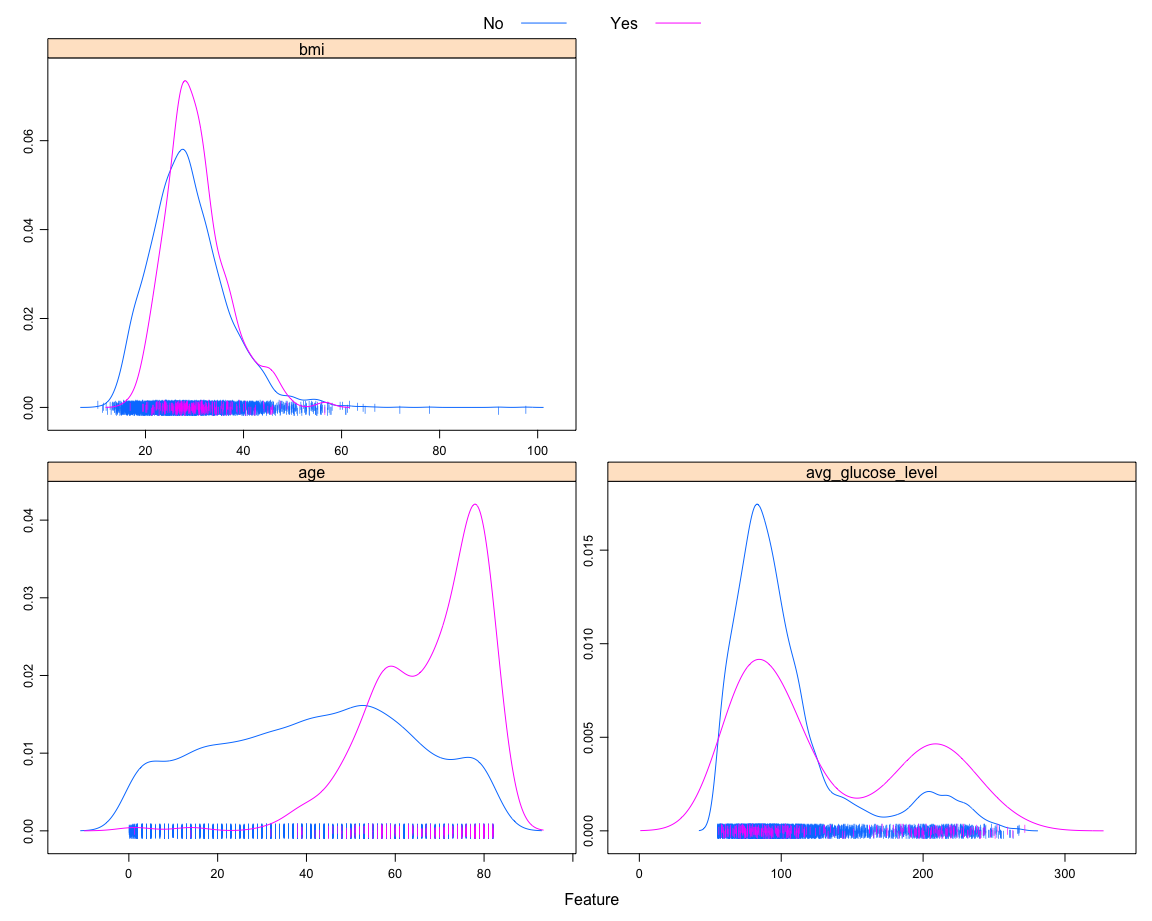
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | Variable | Stats / Values | Freqs (% of Valid) | Graph | Valid | Missing |
| 1 | gender [factor] | 1. Female 2. Male | 2994 (58.6%) 2115 (41.4%) | IIIIIIIIIII IIIIIIII | 5109 (100.0%) | 0 (0.0%) |
| 2 | age [numeric] | Mean (sd) : 43.2 (22.6) min < med < max: 0.1 < 45 < 82 IQR (CV) : 36 (0.5) | 104 distinct values | .   :       . : : : :   : : . : : : : : : : : : : : : : : : : : : : : : : : : : : : : | 5109 (100.0%) | 0 (0.0%) |
| 3 | hypertension [factor] | 1. No 2. Yes | 4611 (90.3%) 498 ( 9.7%) | IIIIIIIIIIIIIIIIII I | 5109 (100.0%) | 0 (0.0%) |
| 4 | heart\_disease [factor] | 1. No 2. Yes | 4833 (94.6%) 276 ( 5.4%) | IIIIIIIIIIIIIIIIII I | 5109 (100.0%) | 0 (0.0%) |
| 5 | ever\_married [factor] | 1. No 2. Yes | 1756 (34.4%) 3353 (65.6%) | IIIIII IIIIIIIIIIIII | 5109 (100.0%) | 0 (0.0%) |
| 6 | work\_type [factor] | 1. children 2. Govt\_job 3. Never\_worked 4. Private 5. Self-employed | 687 (13.4%) 657 (12.9%) 22 ( 0.4%) 2924 (57.2%) 819 (16.0%) | II II  IIIIIIIIIII III | 5109 (100.0%) | 0 (0.0%) |
| 7 | residence\_type [factor] | 1. Rural 2. Urban | 2513 (49.2%) 2596 (50.8%) | IIIIIIIII IIIIIIIIII | 5109 (100.0%) | 0 (0.0%) |
| 8 | avg\_glucose\_level [numeric] | Mean (sd) : 106.1 (45.3) min < med < max: 55.1 < 91.9 < 271.7 IQR (CV) : 36.9 (0.4) | 3978 distinct values | : . : : : . : : : : : : : . . . . . | 5109 (100.0%) | 0 (0.0%) |
| 9 | bmi [numeric] | Mean (sd) : 28.9 (7.9) min < med < max: 10.3 < 28.1 < 97.6 IQR (CV) : 9.6 (0.3) | 418 distinct values | : .   : :   : :   : : . : : : : . | 4908 (96.1%) | 201 (3.9%) |
| 10 | stroke [factor] | 1. No 2. Yes | 4860 (95.1%) 249 ( 4.9%) | IIIIIIIIIIIIIIIIIII | 5109 (100.0%) | 0 (0.0%) |

# Exploratory data analysis

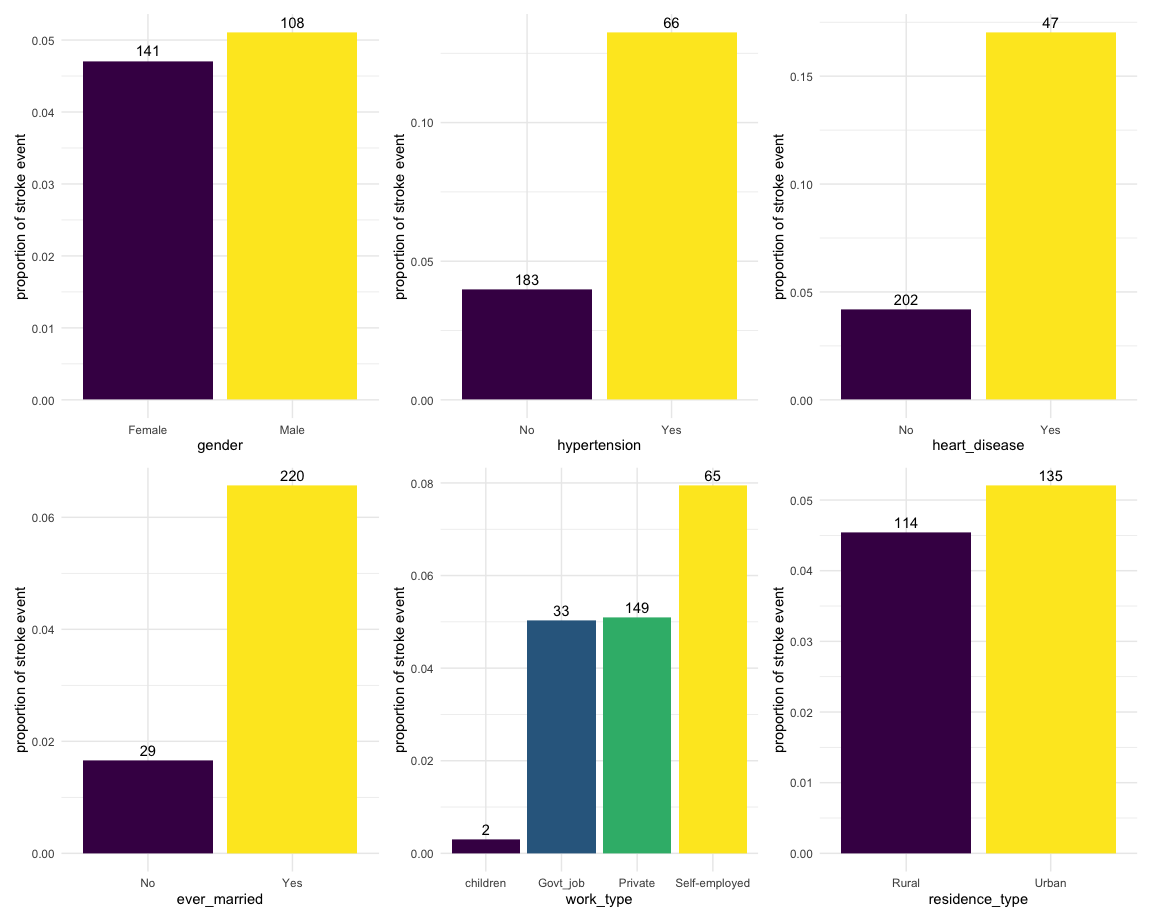
# Correlation matrix among predictors  
stroke\_eda =   
 stroke\_clean %>%   
 mutate(work\_type = relevel(work\_type, ref = "Govt\_job"))  
whole\_x = model.matrix(stroke ~ ., stroke\_eda)[ ,-1]   
corrplot(cor(whole\_x), method = "circle", type = "full")



# Having ever been married seems to be associated with older age. No other apparent correlation between predictors.  
  
# Density plots of continuous predictors across levels of “stroke”   
featurePlot(  
 x = stroke\_clean[ , c(2, 8, 9)],   
 y = stroke\_clean$stroke,  
 scales = list(  
 x = list(relation = "free"),   
 y = list(relation = "free")  
 ),  
 plot = "density", pch = "|",   
 auto.key = list(columns = 2)  
)



# Bar charts between `stroke` and categorical predictors  
gender\_bar =   
 stroke\_clean %>%   
 count(gender, stroke) %>%   
 group\_by(gender) %>%   
 mutate(stroke\_prop = n / sum(n)) %>%   
 filter(stroke == "Yes") %>%   
 ggplot(aes(x = gender, y = stroke\_prop, fill = gender)) +  
 geom\_col() +  
 geom\_text(aes(label = n), vjust = -0.5) +  
 labs(  
 y = "proportion of stroke event"  
 )  
  
hp\_bar =   
 stroke\_clean %>%   
 count(hypertension, stroke) %>%   
 group\_by(hypertension) %>%   
 mutate(stroke\_prop = n / sum(n)) %>%   
 filter(stroke == "Yes") %>%   
 ggplot(aes(x = hypertension, y = stroke\_prop, fill = hypertension)) +  
 geom\_col() +  
 geom\_text(aes(label = n), vjust = -0.5) +  
 labs(  
 y = "proportion of stroke event"  
 )  
  
hd\_bar =   
 stroke\_clean %>%   
 count(heart\_disease, stroke) %>%   
 group\_by(heart\_disease) %>%   
 mutate(stroke\_prop = n / sum(n)) %>%   
 filter(stroke == "Yes") %>%   
 ggplot(aes(x = heart\_disease, y = stroke\_prop, fill = heart\_disease)) +  
 geom\_col() +  
 geom\_text(aes(label = n), vjust = -0.5) +  
 labs(  
 y = "proportion of stroke event"  
 )  
  
married\_bar =   
 stroke\_clean %>%   
 count(ever\_married, stroke) %>%   
 group\_by(ever\_married) %>%   
 mutate(stroke\_prop = n / sum(n)) %>%   
 filter(stroke == "Yes") %>%   
 ggplot(aes(x = ever\_married, y = stroke\_prop, fill = ever\_married)) +  
 geom\_col() +  
 geom\_text(aes(label = n), vjust = -0.5) +  
 labs(  
 y = "proportion of stroke event"  
 )  
  
work\_bar =   
 stroke\_clean %>%   
 count(work\_type, stroke) %>%   
 group\_by(work\_type) %>%   
 mutate(stroke\_prop = n / sum(n)) %>%   
 filter(stroke == "Yes") %>%   
 ggplot(aes(x = work\_type, y = stroke\_prop, fill = work\_type)) +  
 geom\_col() +  
 geom\_text(aes(label = n), vjust = -0.5) +  
 labs(  
 y = "proportion of stroke event"  
 )  
  
residence\_bar =   
 stroke\_clean %>%   
 count(residence\_type, stroke) %>%   
 group\_by(residence\_type) %>%   
 mutate(stroke\_prop = n / sum(n)) %>%   
 filter(stroke == "Yes") %>%   
 ggplot(aes(x = residence\_type, y = stroke\_prop, fill = residence\_type)) +  
 geom\_col() +  
 geom\_text(aes(label = n), vjust = -0.5) +  
 labs(  
 y = "proportion of stroke event"  
 )  
  
(gender\_bar + hp\_bar + hd\_bar) / (married\_bar + work\_bar + residence\_bar)



# Models

# Exclude `work\_type`  
stroke\_nowork =   
 stroke\_clean %>%   
 select(-work\_type)  
  
# Data partition  
set.seed(1)  
train\_indices = createDataPartition(  
 y = stroke\_nowork$stroke,  
 p = 0.75,  
 list = FALSE  
)  
  
# Imputation  
set.seed(1)  
stroke\_x = stroke\_nowork[train\_indices, -9]  
bagimp = preProcess(stroke\_x, method = "bagImpute")  
stroke\_train = predict(bagimp, stroke\_nowork[train\_indices, ])  
stroke\_test = predict(bagimp, stroke\_nowork[-train\_indices, ])  
  
# Train control  
ctrl = trainControl(  
 method = "cv",   
 number = 10,   
 sampling = "down",   
 summaryFunction = twoClassSummary,   
 classProbs = TRUE  
)  
  
# Model 1: LASSO  
set.seed(1)  
lasso\_mod = train(  
 stroke ~ .,  
 data = stroke\_train,  
 method = "glmnet",  
 metric = "ROC",  
 trControl = ctrl,  
 tuneGrid = expand.grid(alpha = 1,   
 lambda = exp(seq(-2, -7, length = 100)))  
)  
  
lasso\_mod$bestTune

alpha lambda 71 1 0.03128406

# Model 2: MARS  
set.seed(1)  
mars\_mod = train(  
 stroke ~ .,  
 data = stroke\_train,  
 method = "earth",  
 tuneGrid = expand.grid(degree = 1:3,   
 nprune = 2:15),  
 metric = "ROC",  
 trControl = ctrl  
)  
  
mars\_mod$bestTune

nprune degree 2 3 1

# Model 3: random forest  
set.seed(1)  
rf\_mod = train(  
 stroke ~ .,  
 data = stroke\_train,  
 method = "ranger",  
 tuneGrid = expand.grid(  
 mtry = 1:8,  
 splitrule = "gini",  
 min.node.size = 1:6  
 ),  
 metric = "ROC",  
 trControl = ctrl  
)  
  
rf\_mod$bestTune

mtry splitrule min.node.size 23 4 gini 5

# Model 4: AdaBoost  
set.seed(1)  
gbm\_mod = train(  
 stroke ~ .,  
 data = stroke\_train,  
 method = "gbm",  
 distribution = "adaboost",  
 tuneGrid = expand.grid(n.trees = c(500, 1000, 2000),  
 interaction.depth = 1:3,  
 shrinkage = c(0.001, 0.003, 0.005),  
 n.minobsinnode = 1:6),  
 metric = "ROC",  
 trControl = ctrl,  
 verbose = FALSE  
)  
  
gbm\_mod$bestTune

n.trees interaction.depth shrinkage n.minobsinnode

136 500 2 0.005 4

## Comparison

comparison =   
 resamples(  
 list(  
 LASSO = lasso\_mod,  
 MARS = mars\_mod,  
 random\_forest = rf\_mod,  
 adaboost = gbm\_mod  
 )  
 )  
  
summary(comparison)

Call: summary.resamples(object = comparison)

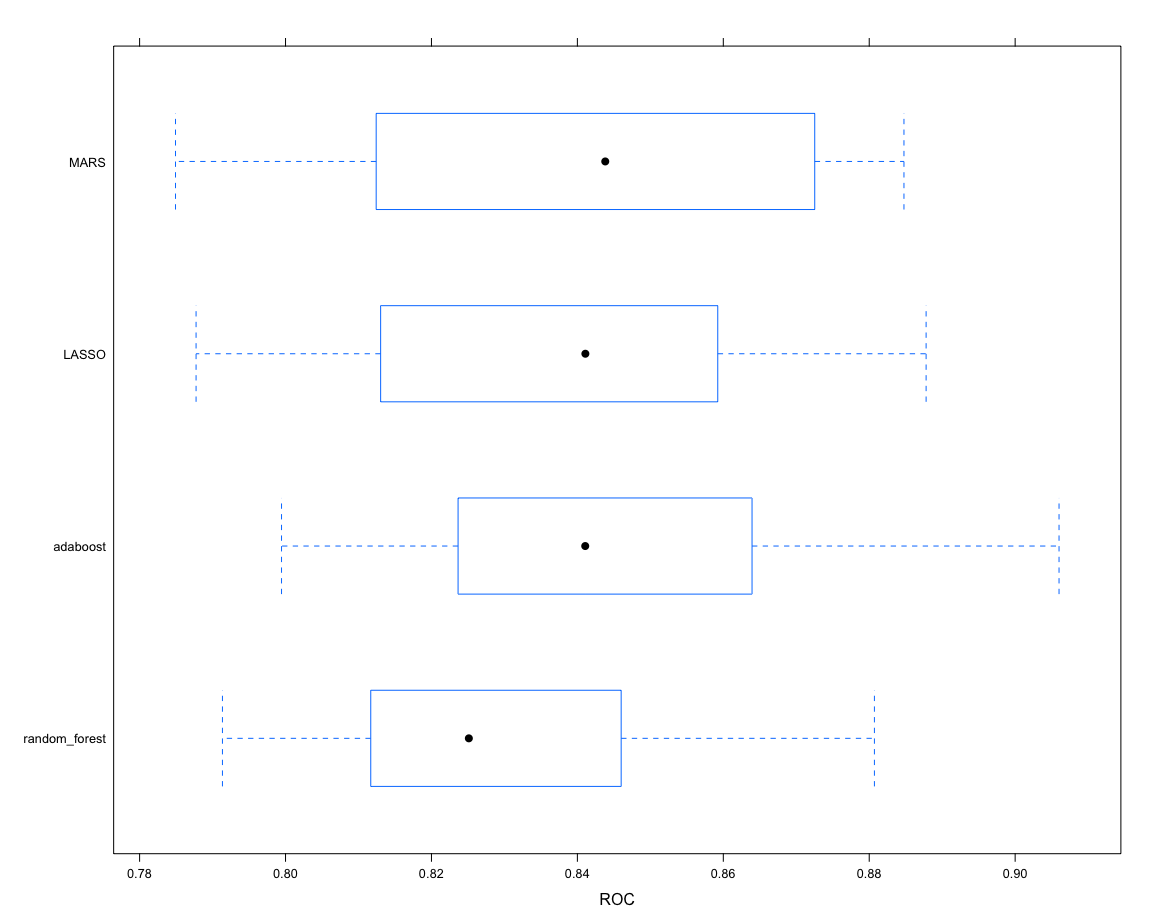
Models: LASSO, MARS, random\_forest, adaboost Number of resamples: 10

ROC Min. 1st Qu. Median Mean 3rd Qu. Max. NA’s LASSO 0.7877386 0.8184895 0.8410959 0.8392834 0.8575706 0.8877964 0 MARS 0.7849190 0.8166618 0.8438213 0.8412711 0.8672314 0.8847600 0 random\_forest 0.7913534 0.8135164 0.8251255 0.8310838 0.8440179 0.8807114 0 adaboost 0.7994505 0.8240807 0.8410759 0.8450025 0.8614766 0.9060150 0

Sens Min. 1st Qu. Median Mean 3rd Qu. Max. NA’s LASSO 0.6602740 0.7012363 0.7201641 0.7160635 0.7363014 0.7582418 0 MARS 0.7060440 0.7278997 0.7410959 0.7410191 0.7554945 0.7692308 0 random\_forest 0.6840659 0.7025384 0.7109589 0.7177096 0.7265204 0.7884615 0 adaboost 0.6547945 0.6684837 0.6978022 0.6963074 0.7141803 0.7609890 0

Spec Min. 1st Qu. Median Mean 3rd Qu. Max. NA’s LASSO 0.6111111 0.7258772 0.8654971 0.8222222 0.8947368 1.0000000 0 MARS 0.6111111 0.7258772 0.7894737 0.8011696 0.8932749 0.9473684 0 random\_forest 0.6315789 0.6937135 0.7850877 0.7853801 0.8947368 0.8947368 0 adaboost 0.6842105 0.7807018 0.8377193 0.8546784 0.9342105 1.0000000 0

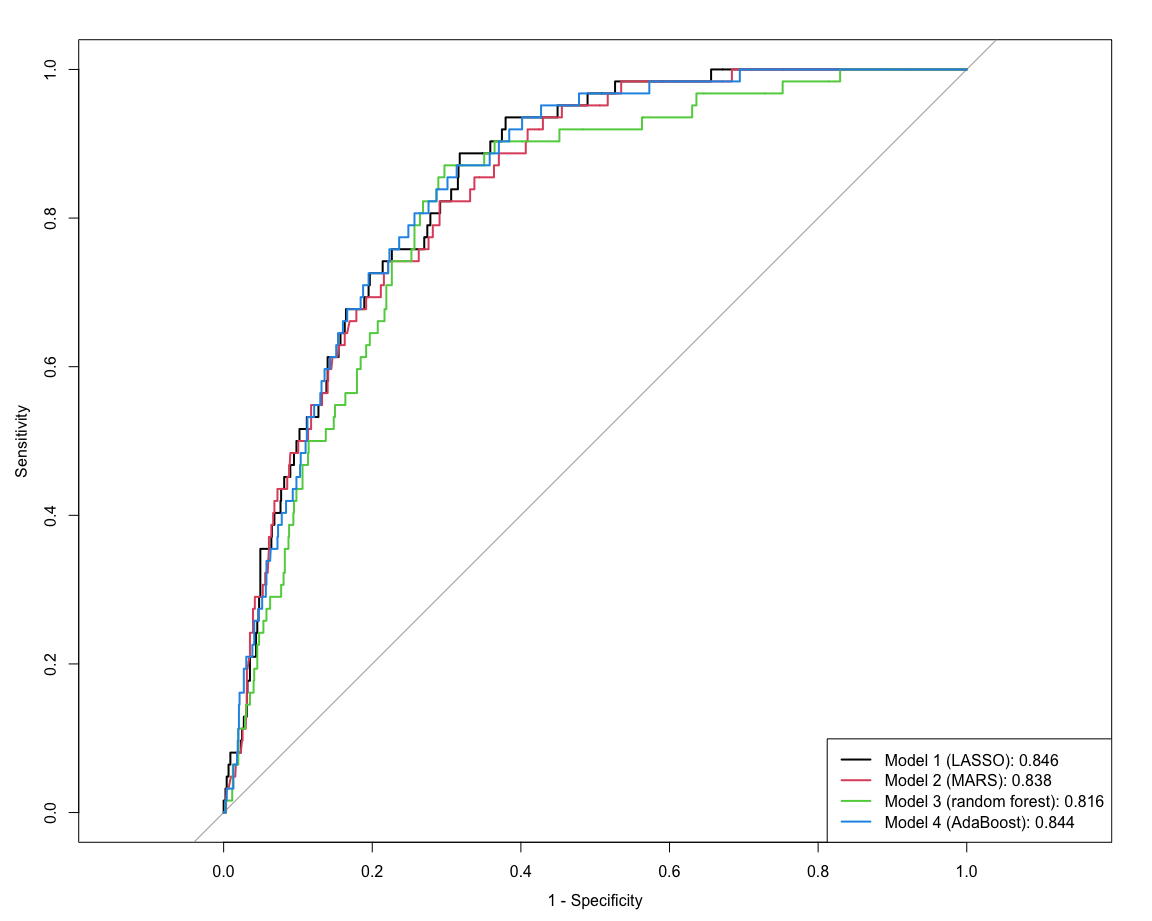
bwplot(comparison, metric = "ROC")



# Model 4 (AdaBoost) was selected as the final model

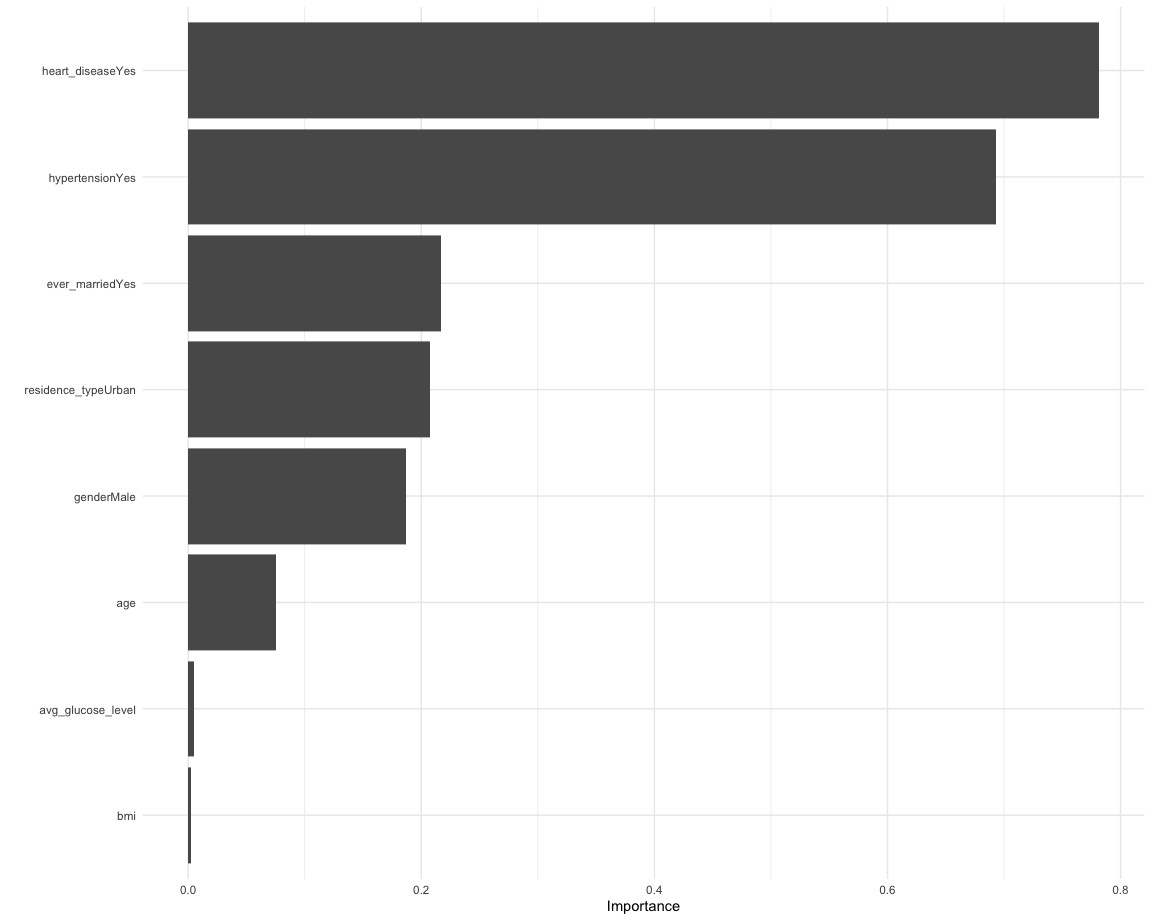
## Test performance

lasso\_pred = predict(lasso\_mod, newdata = stroke\_test, type = "prob")[ , 2]  
mars\_pred = predict(mars\_mod, newdata = stroke\_test, type = "prob")[ , 2]  
rf\_pred = predict(rf\_mod, newdata = stroke\_test, type = "prob")[ , 2]  
gbm\_pred = predict(gbm\_mod, newdata = stroke\_test, type = "prob")[ , 2]  
  
lasso\_roc = roc(stroke\_test$stroke, lasso\_pred)  
mars\_roc = roc(stroke\_test$stroke, mars\_pred)  
rf\_roc = roc(stroke\_test$stroke, rf\_pred)  
gbm\_roc = roc(stroke\_test$stroke, gbm\_pred)  
  
auc\_mod = c(lasso\_roc$auc[1], mars\_roc$auc[1], rf\_roc$auc[1], gbm\_roc$auc[1])  
  
plot(lasso\_roc, col = 1, legacy.axes = TRUE)  
plot(mars\_roc, col = 2, add = TRUE)  
plot(rf\_roc, col = 3, add = TRUE)  
plot(gbm\_roc, col = 4, add = TRUE)  
  
mod\_names = c("Model 1 (LASSO)", "Model 2 (MARS)", "Model 3 (random forest)", "Model 4 (AdaBoost)")  
legend("bottomright", legend = paste0(mod\_names, ": ", round(auc\_mod, 3)), col = 1:5, lwd = 2)

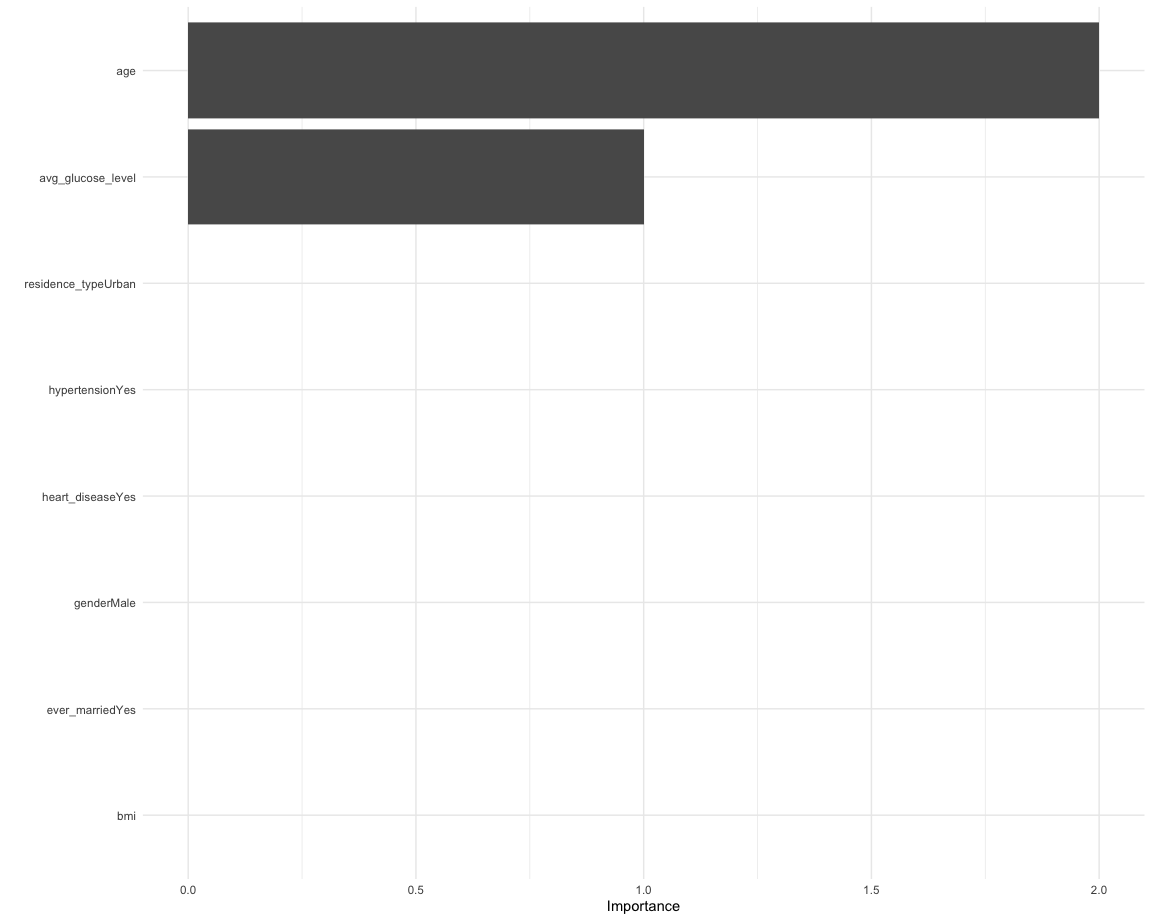


## Variable importance

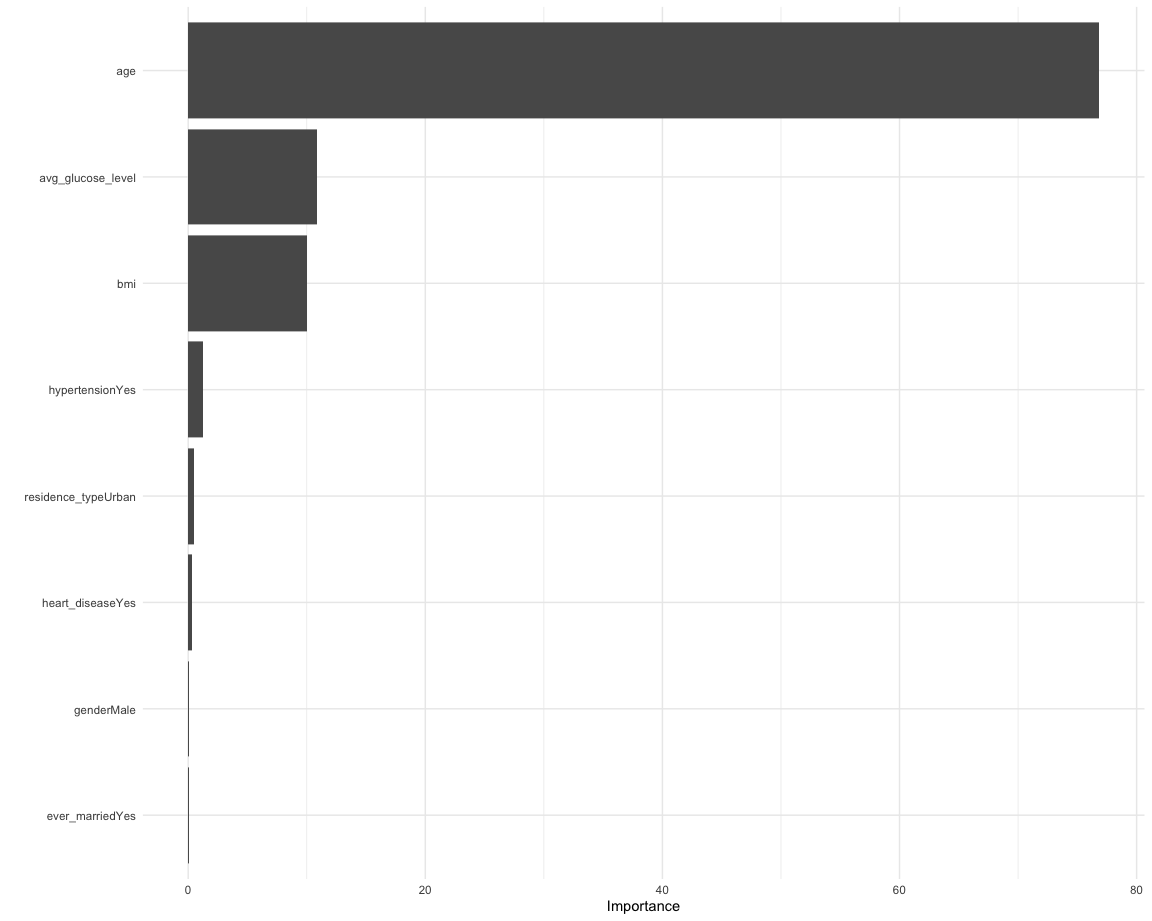
vip(lasso\_mod$finalModel)



vip(mars\_mod$finalModel)



vip(gbm\_mod$finalModel)



set.seed(1)  
rf\_imp = ranger(  
 stroke ~ .,   
 data = stroke\_train,   
 mtry = rf\_mod$bestTune[[1]],   
 splitrule = "gini",  
 min.node.size = rf\_mod$bestTune[[3]],  
 importance = "impurity"  
)   
  
barplot(sort(ranger::importance(rf\_imp), decreasing = FALSE),   
 las = 2, horiz = TRUE, cex.names = 0.7,  
 col = colorRampPalette(colors = c("cyan", "blue"))(8))

