HealthCareAI

Parinitha Kompala

8/26/2021

library(healthcareai)

## healthcareai version 2.5.0  
## Please visit https://docs.healthcare.ai for full documentation and vignettes. Join the community at https://healthcare-ai.slack.com

analysing the pima data

pima\_diabetes #inbuit dataframe

## # A tibble: 768 × 10  
## patient\_id pregnancies plasma\_glucose diastolic\_bp skinfold insulin  
## <int> <int> <int> <int> <int> <int>  
## 1 1 6 148 72 35 NA  
## 2 2 1 85 66 29 NA  
## 3 3 8 183 64 NA NA  
## 4 4 1 89 66 23 94  
## 5 5 0 137 40 35 168  
## 6 6 5 116 74 NA NA  
## 7 7 3 78 50 32 88  
## 8 8 10 115 NA NA NA  
## 9 9 2 197 70 45 543  
## 10 10 8 125 96 NA NA  
## # … with 758 more rows, and 4 more variables: weight\_class <chr>,  
## # pedigree <dbl>, age <int>, diabetes <chr>

dim(pima\_diabetes)

## [1] 768 10

summary(pima\_diabetes)

## patient\_id pregnancies plasma\_glucose diastolic\_bp   
## Min. : 1.0 Min. : 0.000 Min. : 44.0 Min. : 24.00   
## 1st Qu.:192.8 1st Qu.: 1.000 1st Qu.: 99.0 1st Qu.: 64.00   
## Median :384.5 Median : 3.000 Median :117.0 Median : 72.00   
## Mean :384.5 Mean : 3.845 Mean :121.7 Mean : 72.41   
## 3rd Qu.:576.2 3rd Qu.: 6.000 3rd Qu.:141.0 3rd Qu.: 80.00   
## Max. :768.0 Max. :17.000 Max. :199.0 Max. :122.00   
## NA's :5 NA's :35   
## skinfold insulin weight\_class pedigree   
## Min. : 7.00 Min. : 14.00 Length:768 Min. :0.0780   
## 1st Qu.:22.00 1st Qu.: 76.25 Class :character 1st Qu.:0.2437   
## Median :29.00 Median :125.00 Mode :character Median :0.3725   
## Mean :29.15 Mean :155.55 Mean :0.4719   
## 3rd Qu.:36.00 3rd Qu.:190.00 3rd Qu.:0.6262   
## Max. :99.00 Max. :846.00 Max. :2.4200   
## NA's :227 NA's :374   
## age diabetes   
## Min. :21.00 Length:768   
## 1st Qu.:24.00 Class :character   
## Median :29.00 Mode :character   
## Mean :33.24   
## 3rd Qu.:41.00   
## Max. :81.00   
##

str(pima\_diabetes)

## tibble [768 × 10] (S3: tbl\_df/tbl/data.frame)  
## $ patient\_id : int [1:768] 1 2 3 4 5 6 7 8 9 10 ...  
## $ pregnancies : int [1:768] 6 1 8 1 0 5 3 10 2 8 ...  
## $ plasma\_glucose: int [1:768] 148 85 183 89 137 116 78 115 197 125 ...  
## $ diastolic\_bp : int [1:768] 72 66 64 66 40 74 50 NA 70 96 ...  
## $ skinfold : int [1:768] 35 29 NA 23 35 NA 32 NA 45 NA ...  
## $ insulin : int [1:768] NA NA NA 94 168 NA 88 NA 543 NA ...  
## $ weight\_class : chr [1:768] "obese" "overweight" "normal" "overweight" ...  
## $ pedigree : num [1:768] 0.627 0.351 0.672 0.167 2.288 ...  
## $ age : int [1:768] 50 31 32 21 33 30 26 29 53 54 ...  
## $ diabetes : chr [1:768] "Y" "N" "Y" "N" ...

Machine learing models model 1-

head(pima\_diabetes)

## # A tibble: 6 × 10  
## patient\_id pregnancies plasma\_glucose diastolic\_bp skinfold insulin  
## <int> <int> <int> <int> <int> <int>  
## 1 1 6 148 72 35 NA  
## 2 2 1 85 66 29 NA  
## 3 3 8 183 64 NA NA  
## 4 4 1 89 66 23 94  
## 5 5 0 137 40 35 168  
## 6 6 5 116 74 NA NA  
## # … with 4 more variables: weight\_class <chr>, pedigree <dbl>, age <int>,  
## # diabetes <chr>

names(pima\_diabetes)

## [1] "patient\_id" "pregnancies" "plasma\_glucose" "diastolic\_bp"   
## [5] "skinfold" "insulin" "weight\_class" "pedigree"   
## [9] "age" "diabetes"

quick\_models <- machine\_learn(pima\_diabetes, patient\_id, outcome = diabetes)

## Training new data prep recipe...

## Variable(s) ignored in prep\_data won't be used to tune models: patient\_id

##   
## diabetes looks categorical, so training classification algorithms.

##   
## After data processing, models are being trained on 12 features with 768 observations.  
## Based on n\_folds = 5 and hyperparameter settings, the following number of models will be trained: 50 rf's, 50 xgb's, and 100 glm's

## Training with cross validation: Random Forest

## Training with cross validation: eXtreme Gradient Boosting

## Training with cross validation: glmnet

##   
## \*\*\* Models successfully trained. The model object contains the training data minus ignored ID columns. \*\*\*  
## \*\*\* If there was PHI in training data, normal PHI protocols apply to the model object. \*\*\*

quick\_models

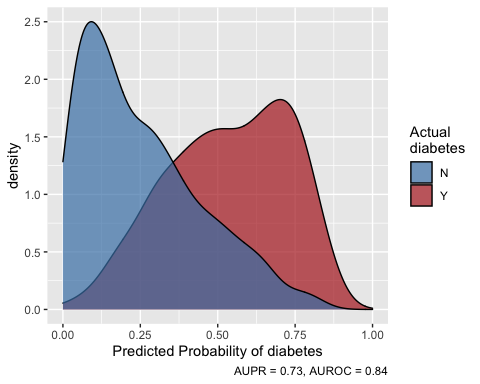
## Algorithms Trained: Random Forest, eXtreme Gradient Boosting, and glmnet  
## Model Name: diabetes  
## Target: diabetes  
## Class: Classification  
## Performance Metric: AUROC  
## Number of Observations: 768  
## Number of Features: 12  
## Models Trained: 2021-08-27 00:28:16   
##   
## Models tuned via 5-fold cross validation over 9 combinations of hyperparameter values.  
## Best model: Random Forest  
## AUPR = 0.72, AUROC = 0.85  
## Optimal hyperparameter values:  
## mtry = 3  
## splitrule = extratrees  
## min.node.size = 5

predictions <- predict(quick\_models)  
predictions

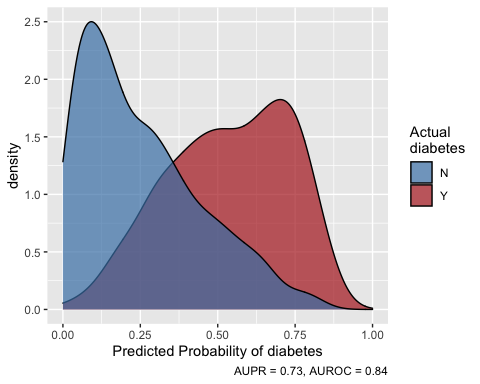
## "predicted\_diabetes" predicted by Random Forest last trained: 2021-08-27 00:28:16  
## Performance in training: AUROC = 0.85

## # A tibble: 768 × 11  
## diabetes predicted\_diabetes patient\_id pregnancies plasma\_glucose diastolic\_bp  
## \* <fct> <dbl> <int> <int> <int> <int>  
## 1 Y 0.619 1 6 148 72  
## 2 N 0.135 2 1 85 66  
## 3 Y 0.398 3 8 183 64  
## 4 N 0.0251 4 1 89 66  
## 5 Y 0.537 5 0 137 40  
## 6 N 0.176 6 5 116 74  
## 7 Y 0.0881 7 3 78 50  
## 8 N 0.538 8 10 115 NA  
## 9 Y 0.741 9 2 197 70  
## 10 Y 0.377 10 8 125 96  
## # … with 758 more rows, and 5 more variables: skinfold <int>, insulin <int>,  
## # weight\_class <chr>, pedigree <dbl>, age <int>

plot(predictions)

 plotting the output od prediction 1

plot(predictions)

 prediction2-

a\_quick\_models <- machine\_learn(pima\_diabetes, plasma\_glucose, outcome = diabetes)

## Warning in prep\_data(d, !!!dots, outcome = !!outcome, impute = impute): These  
## ignored variables have missingness: plasma\_glucose

## Training new data prep recipe...

## Variable(s) ignored in prep\_data won't be used to tune models: plasma\_glucose

##   
## diabetes looks categorical, so training classification algorithms.

##   
## After data processing, models are being trained on 12 features with 768 observations.  
## Based on n\_folds = 5 and hyperparameter settings, the following number of models will be trained: 50 rf's, 50 xgb's, and 100 glm's

## Training with cross validation: Random Forest

## Training with cross validation: eXtreme Gradient Boosting

## Training with cross validation: glmnet

##   
## \*\*\* Models successfully trained. The model object contains the training data minus ignored ID columns. \*\*\*  
## \*\*\* If there was PHI in training data, normal PHI protocols apply to the model object. \*\*\*

a\_quick\_models

## Algorithms Trained: Random Forest, eXtreme Gradient Boosting, and glmnet  
## Model Name: diabetes  
## Target: diabetes  
## Class: Classification  
## Performance Metric: AUROC  
## Number of Observations: 768  
## Number of Features: 12  
## Models Trained: 2021-08-27 00:28:28   
##   
## Models tuned via 5-fold cross validation over 10 combinations of hyperparameter values.  
## Best model: eXtreme Gradient Boosting  
## AUPR = 0.61, AUROC = 0.78  
## Optimal hyperparameter values:  
## nrounds = 640  
## max\_depth = 2  
## eta = 0.0068  
## gamma = 1.9  
## colsample\_bytree = 0.89  
## min\_child\_weight = 5.7  
## subsample = 0.64

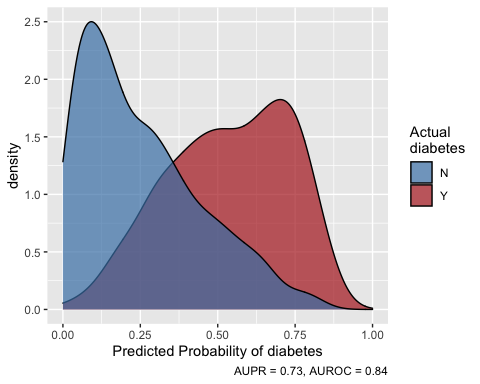
a\_predictions <- predict(quick\_models)  
a\_predictions

## "predicted\_diabetes" predicted by Random Forest last trained: 2021-08-27 00:28:16  
## Performance in training: AUROC = 0.85

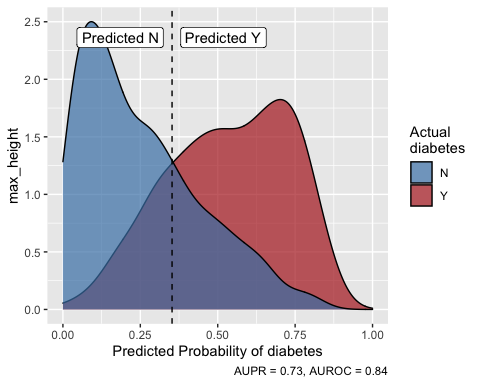
## # A tibble: 768 × 11  
## diabetes predicted\_diabetes patient\_id pregnancies plasma\_glucose diastolic\_bp  
## \* <fct> <dbl> <int> <int> <int> <int>  
## 1 Y 0.619 1 6 148 72  
## 2 N 0.135 2 1 85 66  
## 3 Y 0.398 3 8 183 64  
## 4 N 0.0251 4 1 89 66  
## 5 Y 0.537 5 0 137 40  
## 6 N 0.176 6 5 116 74  
## 7 Y 0.0881 7 3 78 50  
## 8 N 0.538 8 10 115 NA  
## 9 Y 0.741 9 2 197 70  
## 10 Y 0.377 10 8 125 96  
## # … with 758 more rows, and 5 more variables: skinfold <int>, insulin <int>,  
## # weight\_class <chr>, pedigree <dbl>, age <int>

plot of prediction2

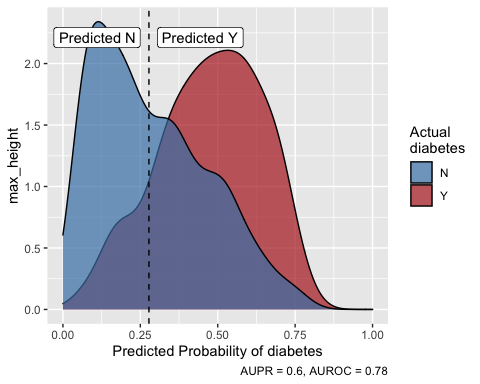
plot(a\_predictions)



quick\_models %>%  
 predict(outcome\_groups = 2) %>%  
 plot()



a\_quick\_models %>%  
 predict(outcome\_groups = 2) %>%  
 plot()

 Doing some data profiling

missingness(pima\_diabetes)  
missingness(pima\_diabetes) %>%  
 plot()

Data preparation

prep\_data

## function (d, ..., outcome, recipe = NULL, remove\_near\_zero\_variance = TRUE,   
## convert\_dates = TRUE, impute = TRUE, collapse\_rare\_factors = TRUE,   
## PCA = FALSE, center = FALSE, scale = FALSE, make\_dummies = TRUE,   
## add\_levels = TRUE, logical\_to\_numeric = TRUE, factor\_outcome = TRUE,   
## no\_prep = FALSE)   
## {  
## if (!is.data.frame(d))   
## stop("\"d\" must be a data frame.")  
## orig\_data <- d  
## new\_recipe <- TRUE  
## if (!is.null(recipe)) {  
## new\_recipe <- FALSE  
## recipe <- check\_rec\_obj(recipe)  
## no\_prep <- attr(recipe, "no\_prep")  
## }  
## if (no\_prep)   
## remove\_near\_zero\_variance <- convert\_dates <- impute <- collapse\_rare\_factors <- center <- scale <- make\_dummies <- add\_levels <- logical\_to\_numeric <- factor\_outcome <- FALSE  
## d\_missing <- missingness(d, return\_df = FALSE)  
## d\_ods <- d[0, ]  
## d\_levels <- get\_factor\_levels(d)  
## best\_levels <- attr(d, "best\_levels")  
## outcome <- rlang::enquo(outcome)  
## remove\_outcome <- FALSE  
## ignore\_columns <- rlang::quos(...)  
## ignored <- purrr::map\_chr(ignore\_columns, rlang::quo\_name)  
## d\_ignore <- NULL  
## if (length(ignored)) {  
## present <- ignored %in% names(d)  
## if (any(!present))   
## stop(list\_variables(ignored[!present]), " not found in d.")  
## if (length(ignored) >= ncol(d))   
## stop("You only have ignored columns. Try again.")  
## d\_ignore <- dplyr::select(d, !!ignored)  
## d <- dplyr::select(d, -dplyr::one\_of(ignored))  
## m <- missingness(d\_ignore) %>% dplyr::filter(percent\_missing >   
## 0)  
## if (!purrr::is\_empty(m$variable))   
## warning("These ignored variables have missingness: ",   
## list\_variables(m$variable))  
## }  
## opt <- options("contrasts")[[1]][[1]]  
## if (opt != "contr.treatment") {  
## w <- paste0("Your unordered-factor contrasts option is set to ",   
## opt, ". This may produce unexpected behavior, particularly in make\_dummies in prep\_data. ",   
## "Consider resetting it by restarting R, or with: ",   
## "options(contrasts = c(\"contr.treatment\", \"contr.poly\"))")  
## warning(w)  
## }  
## if (!new\_recipe) {  
## message("Prepping data based on provided recipe")  
## newvars <- setdiff(names(d), c(recipe$var\_info$variable,   
## attr(recipe, "ignored\_columns")))  
## if (length(newvars)) {  
## warning("These variables were not observed in training ",   
## "and will be ignored: ", list\_variables(newvars))  
## ignored <- c(ignored, newvars)  
## d\_ignore <- dplyr::bind\_cols(d\_ignore, dplyr::select(d,   
## !!newvars))  
## }  
## missing\_vars <- setdiff(recipe$var\_info$variable[recipe$var\_info$role ==   
## "predictor"], names(d))  
## if (length(missing\_vars))   
## warning("These variables were present in training but are missing or ignored here: ",   
## list\_variables(missing\_vars))  
## if (!is.null(recipe$steps)) {  
## if (attr(recipe$steps[[1]], "class")[1] == "step\_nzv") {  
## if (length(recipe$steps[[1]]$removals)) {  
## missing\_vars <- missing\_vars[missing\_vars ==   
## length(recipe$steps[[1]]$removals)]  
## }  
## }  
## }  
## if (length(missing\_vars))   
## stop("These variables were present in training but are missing or ignored here: ",   
## list\_variables(missing\_vars))  
## newly\_missing <- find\_new\_missingness(d, recipe)  
## if (length(newly\_missing))   
## warning("The following variable(s) have missingness that was not present when recipe was trained: ",   
## list\_variables(newly\_missing))  
## outcome\_var <- recipe$var\_info$variable[recipe$var\_info$role ==   
## "outcome"]  
## if (length(outcome\_var) && !outcome\_var %in% names(d))   
## remove\_outcome <- TRUE  
## }  
## else {  
## undeclared\_ignores <- find\_columns\_to\_ignore(d, c(rlang::quo\_name(outcome),   
## ignored))  
## if (length(undeclared\_ignores)) {  
## warning("The following variable(s) look a lot like identifiers: They are ",   
## "character-type and have a unique value on every row. They will ",   
## "be ignored: ", paste0(undeclared\_ignores, collapse = ", "))  
## ignored <- c(ignored, undeclared\_ignores)  
## d\_ignore <- dplyr::bind\_cols(d\_ignore, d[, names(d) %in%   
## undeclared\_ignores, drop = FALSE])  
## d <- d[, !names(d) %in% undeclared\_ignores, drop = FALSE]  
## }  
## mes <- "Training new data prep recipe"  
## recipe <- recipes::recipe(d, ~.)  
## recipe$orig\_data <- orig\_data  
## if (!rlang::quo\_is\_missing(outcome)) {  
## outcome\_name <- rlang::quo\_name(outcome)  
## if (!outcome\_name %in% names(d))   
## stop(paste(outcome\_name, " not found in d."))  
## outcome\_vec <- dplyr::pull(d, !!outcome)  
## if (is.logical(outcome\_vec) || any(c("TRUE", "FALSE") %in%   
## outcome\_vec))   
## stop("outcome looks logical. Please convert the outcome to character",   
## " with values other than TRUE and FALSE.")  
## if (any(is.na(outcome\_vec)))   
## stop("Found NA values in the outcome column. Clean your data or ",   
## "remove these rows before training a model.")  
## suppressWarnings({  
## recipe <- recipes::update\_role(recipe, !!outcome,   
## new\_role = "outcome")  
## })  
## if (factor\_outcome && all(outcome\_vec %in% 0:1)) {  
## if (!is.numeric(outcome\_vec))   
## stop("factor\_outcome is TRUE, but ", outcome\_name,   
## " is a character", "-type variable with 0s and 1s. Consider making it numeric with ",   
## "`as.numeric(as.character())")  
## recipe <- recipe %>% recipes::step\_bin2factor(all\_outcomes(),   
## levels = c("Y", "N"))  
## }  
## mes <- paste0(mes, "...\n")  
## }  
## else {  
## mes <- paste0(mes, " with no outcome variable specified...\n")  
## }  
## message(mes)  
## freq\_cut <- 49  
## unique\_cut <- 10  
## if (!is.logical(remove\_near\_zero\_variance)) {  
## if (!is.numeric(remove\_near\_zero\_variance))   
## stop("remove\_near\_zero\_variance must be logical or numeric for step\_nzv")  
## if (remove\_near\_zero\_variance < 0 | remove\_near\_zero\_variance >   
## 1)   
## stop("remove\_near\_zero\_variance must be numeric between 0 and 1")  
## freq\_cut <- remove\_near\_zero\_variance^-1  
## remove\_near\_zero\_variance <- TRUE  
## }  
## if (remove\_near\_zero\_variance) {  
## recipe <- recipe %>% recipes::step\_nzv(all\_predictors(),   
## freq\_cut = freq\_cut, unique\_cut = unique\_cut)  
## }  
## prep\_check <- recipes::prep(recipe, training = d)  
## removing <- prep\_check$steps[[1]]$removals  
## vi <- recipe$var\_info  
## nom\_preds <- vi$variable[vi$role == "predictor" & vi$type ==   
## "nominal"]  
## if (length(nom\_preds) && all(nom\_preds %in% removing))   
## stop("All your categorical columns will be removed because they have ",   
## "near-zero variance, which will break prep\_data. ",   
## "Be less aggressive in removing near-zero variance columns by ",   
## "using a larger value of remove\_near\_zero\_variance or setting it ",   
## "to FALSE.\n ", list\_variables(removing))  
## if (!is.character(convert\_dates)) {  
## if (!is.logical(convert\_dates))   
## stop("convert\_dates must be logical, \"none\", \"continuous\", or ",   
## "\"categories\"")  
## if (convert\_dates)   
## convert\_dates <- "continuous"  
## else convert\_dates <- "none"  
## }  
## if (convert\_dates %in% c("continuous", "categories")) {  
## cols <- find\_date\_cols(d)  
## if (!purrr::is\_empty(cols)) {  
## recipe <- do.call(step\_date\_hcai, list(recipe = recipe,   
## cols, feature\_type = convert\_dates)) %>% recipes::step\_rm(cols)  
## }  
## }  
## else if (convert\_dates == "none") {  
## cols <- find\_date\_cols(d)  
## if (!purrr::is\_empty(cols))   
## recipe <- recipes::step\_rm(recipe, cols)  
## }  
## else {  
## stop("convert\_dates must be logical, \"none\", \"continuous\", or ",   
## "\"categories\"")  
## }  
## if (isTRUE(impute)) {  
## recipe <- recipe %>% hcai\_impute()  
## }  
## else if (is.list(impute)) {  
## ip <- list(numeric\_method = "mean", nominal\_method = "new\_category",   
## numeric\_params = NULL, nominal\_params = NULL)  
## ip[names(ip) %in% names(impute)] <- impute[names(impute) %in%   
## names(ip)]  
## extras <- names(impute)[!(names(impute) %in% names(ip))]  
## if (length(extras > 0)) {  
## warning("You have extra imputation parameters that won't be used: ",   
## list\_variables(extras), ". Available params are: ",   
## list\_variables(names(ip)))  
## }  
## recipe <- recipe %>% hcai\_impute(numeric\_method = ip$numeric\_method,   
## nominal\_method = ip$nominal\_method, numeric\_params = ip$numeric\_params,   
## nominal\_params = ip$nominal\_params)  
## }  
## else if (impute != FALSE) {  
## stop("impute must be boolean or list.")  
## }  
## if (!(is.numeric(PCA) || is.logical(PCA)))   
## stop("PCA must be logical or numeric")  
## if (as.logical(PCA)) {  
## if (!(as.logical(center) && as.logical(scale))) {  
## warning("\"d\" must be centered and scaled to perform PCA. Center and Scale are being set to TRUE.")  
## center <- as.logical(PCA)  
## scale <- as.logical(PCA)  
## }  
## }  
## var\_info <- recipe$var\_info  
## if (any(var\_info$type == "numeric" & var\_info$role ==   
## "predictor")) {  
## if (isTRUE(as.logical(center))) {  
## recipe <- recipe %>% recipes::step\_center(all\_numeric(),   
## -all\_outcomes())  
## }  
## if (isTRUE(as.logical(scale))) {  
## recipe <- recipe %>% recipes::step\_scale(all\_numeric(),   
## -all\_outcomes())  
## }  
## }  
## if (any(var\_info$type == "nominal" & var\_info$role ==   
## "predictor")) {  
## if (add\_levels)   
## recipe <- step\_add\_levels(recipe, all\_nominal(),   
## -all\_outcomes())  
## if (!is.logical(collapse\_rare\_factors)) {  
## if (!is.numeric(collapse\_rare\_factors))   
## stop("collapse\_rare\_factors must be logical or numeric")  
## if (collapse\_rare\_factors >= 1 || collapse\_rare\_factors <   
## 0)   
## stop("If numeric, collapse\_rare\_factors should be between 0 and 1.")  
## fac\_thresh <- collapse\_rare\_factors  
## collapse\_rare\_factors <- TRUE  
## }  
## if (collapse\_rare\_factors) {  
## if (!exists("fac\_thresh"))   
## fac\_thresh <- 0.03  
## recipe <- recipe %>% recipes::step\_other(all\_nominal(),   
## -all\_outcomes(), threshold = fac\_thresh)  
## }  
## if (add\_levels)   
## recipe <- step\_add\_levels(recipe, all\_nominal(),   
## -all\_outcomes())  
## if (isTRUE(make\_dummies)) {  
## make\_dummies <- list()  
## }  
## if (is.list(make\_dummies)) {  
## recipe <- recipe %>% step\_dummy\_hcai(all\_nominal(),   
## -all\_outcomes(), levels = make\_dummies)  
## }  
## else if (!is.logical(make\_dummies)) {  
## stop("step\_dummies must be logical or list")  
## }  
## }  
## if (as.logical(PCA)) {  
## if (!impute && !is.list(impute))   
## stop("NAs present in \"d\". PCA not compatible when NAs are present.")  
## if (is.logical(PCA))   
## PCA <- 5  
## if (PCA > length(recipes::prep(recipe, training = d)$term\_info$role ==   
## "predictor"))   
## stop("Can't have more components than columns in \"d\".")  
## recipe <- recipe %>% recipes::step\_pca(all\_numeric(),   
## -all\_outcomes(), num\_comp = as.integer(PCA))  
## }  
## recipe <- recipes::prep(recipe, training = d)  
## attr(recipe, "missingness") <- d\_missing  
## attr(recipe, "factor\_levels") <- d\_levels  
## }  
## if (logical\_to\_numeric)   
## d <- dplyr::mutate\_if(d, is.logical, as.numeric)  
## d <- recipes::bake(recipe, d)  
## steps <- map\_chr(recipe$steps, ~attr(.x, "class")[1])  
## if ("step\_nzv" %in% steps && length(nzv\_removed <- recipe$steps[[which(steps ==   
## "step\_nzv")]]$removals))   
## message("Removing the following ", length(nzv\_removed),   
## " near-zero variance column(s). ", "If you don't want to remove them, call prep\_data with ",   
## "remove\_near\_zero\_variance as a smaller numeric or FALSE.\n ",   
## list\_variables(nzv\_removed))  
## if (remove\_outcome && outcome\_var %in% names(d))   
## d <- select\_not(d, outcome\_var)  
## if (rlang::quo\_name(outcome) %in% names(d\_ods))   
## d\_ods <- select\_not(d\_ods, outcome)  
## d <- dplyr::bind\_cols(d\_ignore, d)  
## if (new\_recipe)   
## recipe$template <- dplyr::bind\_cols(d\_ignore, recipe$template)  
## attr(recipe, "ignored\_columns") <- unname(ignored)  
## attr(recipe, "no\_prep") <- no\_prep  
## attr(d, "recipe") <- recipe  
## attr(d, "best\_levels") <- best\_levels  
## attr(d, "original\_data\_str") <- d\_ods  
## d <- tibble::as\_tibble(d)  
## class(d) <- c("prepped\_df", class(d))  
## return(d)  
## }  
## <bytecode: 0x7faf26546dc8>  
## <environment: namespace:healthcareai>

split\_data <- split\_train\_test(d = pima\_diabetes,  
 outcome = diabetes,  
 p = .9,  
 seed = 84105)  
prepped\_training\_data <- prep\_data(split\_data$train, patient\_id, outcome = diabetes,  
 center = TRUE, scale = TRUE,  
 collapse\_rare\_factors = FALSE)

## Training new data prep recipe...

Data training

models <- tune\_models(d = prepped\_training\_data,  
 outcome = diabetes,  
 tune\_depth = 25,  
 metric = "PR")

## Variable(s) ignored in prep\_data won't be used to tune models: patient\_id

##   
## diabetes looks categorical, so training classification algorithms.

##   
## After data processing, models are being trained on 13 features with 692 observations.  
## Based on n\_folds = 5 and hyperparameter settings, the following number of models will be trained: 125 rf's, 125 xgb's, and 250 glm's

## Training with cross validation: Random Forest

## Training with cross validation: eXtreme Gradient Boosting

## Training with cross validation: glmnet

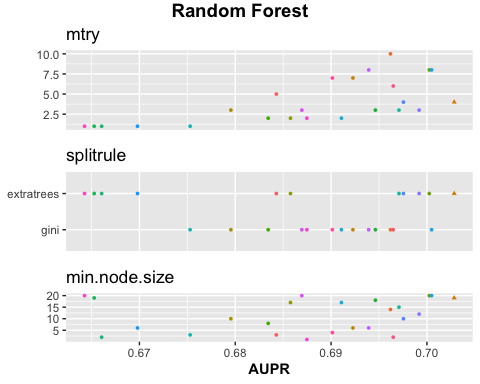
##   
## \*\*\* Models successfully trained. The model object contains the training data minus ignored ID columns. \*\*\*  
## \*\*\* If there was PHI in training data, normal PHI protocols apply to the model object. \*\*\*

evaluate(models, all\_models = TRUE)

## # A tibble: 3 × 3  
## model AUPR AUROC  
## <chr> <dbl> <dbl>  
## 1 Random Forest 0.703 0.842  
## 2 glmnet 0.688 0.836  
## 3 eXtreme Gradient Boosting 0.687 0.820

models["Random Forest"] %>%  
 plot()

## Warning: It is deprecated to specify `guide = FALSE` to remove a guide. Please  
## use `guide = "none"` instead.  
  
## Warning: It is deprecated to specify `guide = FALSE` to remove a guide. Please  
## use `guide = "none"` instead.  
  
## Warning: It is deprecated to specify `guide = FALSE` to remove a guide. Please  
## use `guide = "none"` instead.  
  
## Warning: It is deprecated to specify `guide = FALSE` to remove a guide. Please  
## use `guide = "none"` instead.  
  
## Warning: It is deprecated to specify `guide = FALSE` to remove a guide. Please  
## use `guide = "none"` instead.  
  
## Warning: It is deprecated to specify `guide = FALSE` to remove a guide. Please  
## use `guide = "none"` instead.

 faster madel traing

untuned\_rf <- flash\_models(d = prepped\_training\_data,  
 outcome = diabetes,  
 models = "RF",  
 metric = "PR")

## Variable(s) ignored in prep\_data won't be used to tune models: patient\_id

##   
## diabetes looks categorical, so training classification algorithms.

##   
## After data processing, models are being trained on 13 features with 692 observations.  
## Based on n\_folds = 5 and hyperparameter settings, the following number of models will be trained: 5 rf's

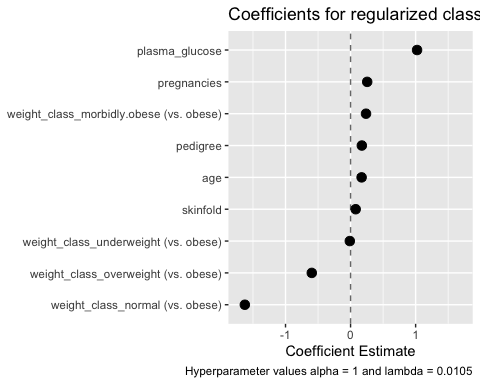
## Training at fixed values: Random Forest

##   
## \*\*\* Models successfully trained. The model object contains the training data minus ignored ID columns. \*\*\*  
## \*\*\* If there was PHI in training data, normal PHI protocols apply to the model object. \*\*\*

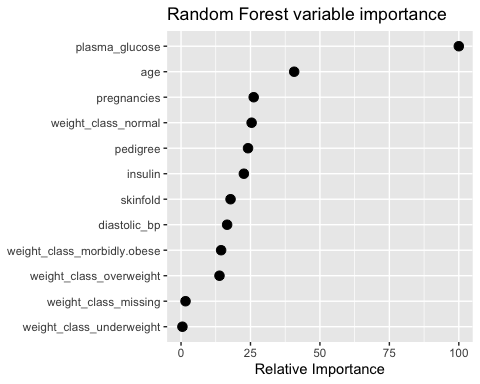
Model interpretations

interpret(models) %>%  
 plot()

## Warning in interpret(models): Interpreting glmnet model, but Random Forest  
## performed best in cross-validation and will be used to make predictions. To use  
## the glmnet model for predictions, extract it with x['glmnet'].

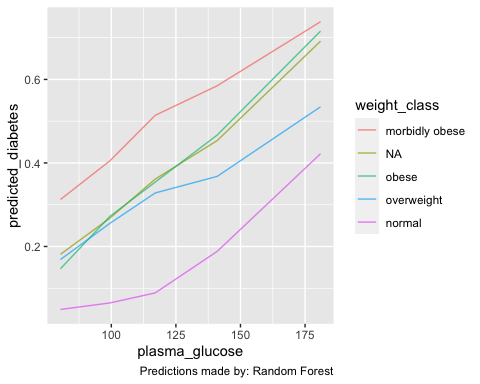
 Variable importance

get\_variable\_importance(models) %>%  
 plot()

 Exploring the data

explore(models) %>%  
 plot()

## With 4 varying features and n\_use = 2, using median to aggregate predicted outcomes across age and pregnancies. You could turn `n\_use` up to see the impact of more features.

 Prediction

predict(models)

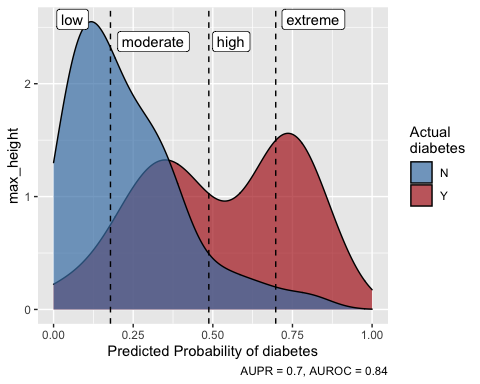
## "predicted\_diabetes" predicted by Random Forest last trained: 2021-08-27 00:29:00  
## Performance in training: AUPR = 0.7

## # A tibble: 692 × 11  
## diabetes predicted\_diabetes patient\_id pregnancies plasma\_glucose diastolic\_bp  
## \* <fct> <dbl> <int> <int> <int> <int>  
## 1 Y 0.691 1 6 148 72  
## 2 N 0.142 2 1 85 66  
## 3 Y 0.432 3 8 183 64  
## 4 N 0.0219 4 1 89 66  
## 5 Y 0.534 5 0 137 40  
## 6 N 0.190 6 5 116 74  
## 7 Y 0.0844 7 3 78 50  
## 8 N 0.398 8 10 115 NA  
## 9 Y 0.776 9 2 197 70  
## 10 Y 0.546 10 8 125 96  
## # … with 682 more rows, and 5 more variables: skinfold <int>, insulin <int>,  
## # weight\_class <chr>, pedigree <dbl>, age <int>

test\_predictions <-  
 predict(models,  
 split\_data$test,  
 risk\_groups = c(low = 30, moderate = 40, high = 20, extreme = 10)  
 )

## Prepping data based on provided recipe

plot(test\_predictions)

 Saving Moving Loading models

save\_models(models, file = "my\_models.RDS")  
models <- load\_models("my\_models.RDS")

regression\_models <- machine\_learn(pima\_diabetes, patient\_id, outcome = age)

## Training new data prep recipe...

## Variable(s) ignored in prep\_data won't be used to tune models: patient\_id

##   
## age looks numeric, so training regression algorithms.

##   
## After data processing, models are being trained on 14 features with 768 observations.  
## Based on n\_folds = 5 and hyperparameter settings, the following number of models will be trained: 50 rf's, 50 xgb's, and 100 glm's

## Training with cross validation: Random Forest

## Training with cross validation: eXtreme Gradient Boosting

## Training with cross validation: glmnet

##   
## \*\*\* Models successfully trained. The model object contains the training data minus ignored ID columns. \*\*\*  
## \*\*\* If there was PHI in training data, normal PHI protocols apply to the model object. \*\*\*

summary(regression\_models)

## Models trained: 2021-08-27 00:29:16  
##   
## Models tuned via 5-fold cross validation over 10 combinations of hyperparameter values.  
## Best performance: RMSE = 9.1, MAE = 6.5, Rsquared = 0.41  
## By Random Forest with hyperparameters:  
## mtry = 4  
## splitrule = variance  
## min.node.size = 17  
##   
## Out-of-fold performance of all trained models:  
##   
## $`Random Forest`  
## # A tibble: 10 × 9  
## mtry splitrule min.node.size RMSE Rsquared MAE RMSESD RsquaredSD MAESD  
## <int> <chr> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 4 variance 17 9.07 0.410 6.53 0.655 0.0283 0.276  
## 2 3 variance 4 9.08 0.412 6.59 0.694 0.0283 0.305  
## 3 5 variance 7 9.09 0.404 6.53 0.601 0.0246 0.235  
## 4 4 extratrees 17 9.16 0.417 6.66 0.814 0.0408 0.441  
## 5 7 variance 2 9.20 0.391 6.62 0.587 0.0209 0.212  
## 6 11 variance 18 9.24 0.385 6.59 0.616 0.0309 0.245  
## 7 2 extratrees 8 9.98 0.388 7.68 0.806 0.0453 0.465  
## 8 2 extratrees 15 9.99 0.388 7.68 0.856 0.0461 0.540  
## 9 2 extratrees 14 10.0 0.381 7.71 0.804 0.0414 0.475  
## 10 1 variance 10 10.5 0.372 8.38 0.711 0.0277 0.431  
##   
## $`eXtreme Gradient Boosting`  
## # A tibble: 10 × 13  
## eta max\_depth gamma colsample\_bytree min\_child\_weight subsample nrounds  
## <dbl> <int> <dbl> <dbl> <dbl> <dbl> <int>  
## 1 0.0291 4 5.73 0.730 0.248 0.763 570  
## 2 0.176 7 9.76 0.518 3.53 0.744 46  
## 3 0.0990 2 0.350 0.624 2.33 0.526 626  
## 4 0.423 5 6.79 0.643 3.80 0.940 69  
## 5 0.432 5 6.23 0.505 14.2 0.356 30  
## 6 0.173 5 2.86 0.849 14.2 0.579 507  
## 7 0.412 6 8.83 0.699 0.673 0.657 642  
## 8 0.495 2 0.155 0.589 4.22 0.597 278  
## 9 0.457 4 0.914 0.556 1.34 0.370 50  
## 10 0.00186 3 8.70 0.699 6.46 0.748 38  
## # … with 6 more variables: RMSE <dbl>, Rsquared <dbl>, MAE <dbl>, RMSESD <dbl>,  
## # RsquaredSD <dbl>, MAESD <dbl>  
##   
## $glmnet  
## # A tibble: 20 × 8  
## alpha lambda RMSE Rsquared MAE RMSESD RsquaredSD MAESD  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 0 0.00128 9.37 0.377 6.74 0.578 0.0798 0.358  
## 2 0 0.00367 9.37 0.377 6.74 0.578 0.0798 0.358  
## 3 0 0.00896 9.37 0.377 6.74 0.578 0.0798 0.358  
## 4 0 0.0218 9.37 0.377 6.74 0.578 0.0798 0.358  
## 5 0 0.0367 9.37 0.377 6.74 0.578 0.0798 0.358  
## 6 0 0.0413 9.37 0.377 6.74 0.578 0.0798 0.358  
## 7 0 0.210 9.37 0.377 6.74 0.578 0.0798 0.358  
## 8 0 0.240 9.37 0.377 6.74 0.578 0.0798 0.358  
## 9 0 0.941 9.37 0.377 6.75 0.565 0.0793 0.342  
## 10 1 0.0413 9.38 0.376 6.72 0.602 0.0801 0.380  
## 11 1 0.0367 9.38 0.376 6.72 0.604 0.0802 0.382  
## 12 1 0.0218 9.38 0.376 6.72 0.609 0.0805 0.387  
## 13 1 0.00128 9.38 0.376 6.72 0.611 0.0807 0.390  
## 14 1 0.00367 9.38 0.376 6.72 0.611 0.0807 0.390  
## 15 1 0.00896 9.38 0.376 6.72 0.611 0.0807 0.390  
## 16 1 0.210 9.40 0.373 6.73 0.566 0.0770 0.330  
## 17 1 0.240 9.40 0.372 6.74 0.563 0.0764 0.323  
## 18 1 0.941 9.50 0.366 6.93 0.539 0.0739 0.248  
## 19 0 6.89 9.58 0.374 7.12 0.502 0.0725 0.217  
## 20 1 6.89 11.7 NaN 9.59 0.549 NA 0.336

Regretion model 2.

a\_regression\_models <- machine\_learn(pima\_diabetes, pregnancies, outcome = age)

## Training new data prep recipe...

## Variable(s) ignored in prep\_data won't be used to tune models: pregnancies

##   
## age looks numeric, so training regression algorithms.

##   
## After data processing, models are being trained on 14 features with 768 observations.  
## Based on n\_folds = 5 and hyperparameter settings, the following number of models will be trained: 50 rf's, 50 xgb's, and 100 glm's

## Training with cross validation: Random Forest

## Training with cross validation: eXtreme Gradient Boosting

## Training with cross validation: glmnet

##   
## \*\*\* Models successfully trained. The model object contains the training data minus ignored ID columns. \*\*\*  
## \*\*\* If there was PHI in training data, normal PHI protocols apply to the model object. \*\*\*

summary(a\_regression\_models)

## Models trained: 2021-08-27 00:29:28  
##   
## Models tuned via 5-fold cross validation over 10 combinations of hyperparameter values.  
## Best performance: RMSE = 10, MAE = 8.1, Rsquared = 0.22  
## By Random Forest with hyperparameters:  
## mtry = 3  
## splitrule = variance  
## min.node.size = 17  
##   
## Out-of-fold performance of all trained models:  
##   
## $`Random Forest`  
## # A tibble: 10 × 9  
## mtry splitrule min.node.size RMSE Rsquared MAE RMSESD RsquaredSD MAESD  
## <int> <chr> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 3 variance 17 10.4 0.224 8.13 0.803 0.0534 0.408  
## 2 5 variance 19 10.4 0.220 8.09 0.770 0.0476 0.403  
## 3 4 variance 10 10.4 0.215 8.13 0.773 0.0478 0.411  
## 4 5 extratrees 5 10.5 0.201 8.20 0.773 0.0261 0.401  
## 5 14 extratrees 10 10.5 0.198 8.18 0.776 0.0248 0.417  
## 6 12 variance 15 10.6 0.201 8.17 0.753 0.0498 0.424  
## 7 13 variance 14 10.6 0.201 8.17 0.775 0.0504 0.434  
## 8 3 extratrees 18 10.6 0.202 8.38 0.823 0.0324 0.354  
## 9 1 extratrees 10 11.4 0.149 9.28 0.781 0.0501 0.343  
## 10 1 extratrees 8 11.4 0.158 9.28 0.783 0.0516 0.345  
##   
## $`eXtreme Gradient Boosting`  
## # A tibble: 10 × 13  
## eta max\_depth gamma colsample\_bytree min\_child\_weight subsample nrounds  
## <dbl> <int> <dbl> <dbl> <dbl> <dbl> <int>  
## 1 0.0387 1 7.68 0.810 0.602 0.380 178  
## 2 0.294 1 9.57 0.747 13.1 0.438 169  
## 3 0.110 2 4.32 0.535 12.4 0.581 629  
## 4 0.180 9 5.92 0.721 9.58 0.992 82  
## 5 0.169 10 4.56 0.563 14.5 0.766 141  
## 6 0.177 4 9.21 0.541 4.34 0.961 398  
## 7 0.157 4 7.46 0.704 2.38 0.833 246  
## 8 0.482 8 3.13 0.700 23.9 0.382 55  
## 9 0.355 10 2.92 0.634 0.271 0.703 458  
## 10 0.491 8 4.74 0.623 2.66 0.674 107  
## # … with 6 more variables: RMSE <dbl>, Rsquared <dbl>, MAE <dbl>, RMSESD <dbl>,  
## # RsquaredSD <dbl>, MAESD <dbl>  
##   
## $glmnet  
## # A tibble: 20 × 8  
## alpha lambda RMSE Rsquared MAE RMSESD RsquaredSD MAESD  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 1 0.172 10.8 0.171 8.43 0.436 0.0517 0.347  
## 2 1 0.372 10.8 0.172 8.45 0.449 0.0544 0.345  
## 3 0 1.27 10.8 0.169 8.44 0.411 0.0470 0.326  
## 4 0 0.872 10.8 0.169 8.43 0.412 0.0475 0.333  
## 5 0 0.826 10.8 0.169 8.43 0.412 0.0475 0.333  
## 6 1 0.0449 10.8 0.170 8.42 0.423 0.0493 0.352  
## 7 0 0.00549 10.8 0.169 8.42 0.415 0.0479 0.341  
## 8 0 0.0124 10.8 0.169 8.42 0.415 0.0479 0.341  
## 9 0 0.0156 10.8 0.169 8.42 0.415 0.0479 0.341  
## 10 0 0.0449 10.8 0.169 8.42 0.415 0.0479 0.341  
## 11 0 0.172 10.8 0.169 8.42 0.415 0.0479 0.341  
## 12 0 0.372 10.8 0.169 8.42 0.415 0.0479 0.341  
## 13 1 0.0156 10.8 0.169 8.41 0.421 0.0487 0.351  
## 14 1 0.0124 10.8 0.169 8.41 0.420 0.0486 0.350  
## 15 1 0.00549 10.8 0.169 8.41 0.420 0.0485 0.350  
## 16 0 5.36 10.8 0.166 8.54 0.431 0.0427 0.288  
## 17 1 0.826 10.8 0.168 8.57 0.454 0.0547 0.312  
## 18 1 0.872 10.8 0.167 8.58 0.456 0.0542 0.310  
## 19 1 1.27 11.0 0.151 8.74 0.490 0.0456 0.292  
## 20 1 5.36 11.7 NaN 9.59 0.579 NA 0.308

#prediction on a hypothetical new patient 1

new\_patient <- data.frame(  
 pregnancies = 0,  
 plasma\_glucose = 80,  
 diastolic\_bp = 55,  
 skinfold = 24,  
 insulin = NA,  
 weight\_class = "???",  
 pedigree = .2,  
 diabetes = "N")  
predict(regression\_models, new\_patient)

## Warning in ready\_with\_prep(object, newdata, mi): The following variables(s) had the following value(s) in predict that were not observed in training.   
## weight\_class: ???

## Prepping data based on provided recipe

## "predicted\_age" predicted by Random Forest last trained: 2021-08-27 00:29:16  
## Performance in training: RMSE = 9.07

## # A tibble: 1 × 9  
## predicted\_age pregnancies plasma\_glucose diastolic\_bp skinfold insulin  
## \* <dbl> <dbl> <dbl> <dbl> <dbl> <lgl>   
## 1 23.7 0 80 55 24 NA   
## # … with 3 more variables: weight\_class <chr>, pedigree <dbl>, diabetes <chr>

#prediction on a hypothetical new patient 2

new\_patient\_2 <- data.frame(  
 pregnancies = 3,  
 plasma\_glucose = 130,  
 diastolic\_bp = 60,  
 skinfold = 24,  
 insulin = NA,  
 weight\_class = "???",  
 pedigree = .2,  
 diabetes = "N")  
predict(regression\_models, new\_patient\_2)

## Warning in ready\_with\_prep(object, newdata, mi): The following variables(s) had the following value(s) in predict that were not observed in training.   
## weight\_class: ???

## Prepping data based on provided recipe

## "predicted\_age" predicted by Random Forest last trained: 2021-08-27 00:29:16  
## Performance in training: RMSE = 9.07

## # A tibble: 1 × 9  
## predicted\_age pregnancies plasma\_glucose diastolic\_bp skinfold insulin  
## \* <dbl> <dbl> <dbl> <dbl> <dbl> <lgl>   
## 1 24.9 3 130 60 24 NA   
## # … with 3 more variables: weight\_class <chr>, pedigree <dbl>, diabetes <chr>