P8451 Machine Learning in Public Health - Assignment 6

2023-2-28

In preparation for all the analyses below, we will load the following libraries:

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──  
## ✔ ggplot2 3.4.1 ✔ purrr 1.0.1  
## ✔ tibble 3.1.8 ✔ dplyr 1.1.0  
## ✔ tidyr 1.3.0 ✔ stringr 1.5.0  
## ✔ readr 2.1.4 ✔ forcats 1.0.0  
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(rpart)  
library(caret)

## Loading required package: lattice  
##   
## Attaching package: 'caret'  
##   
## The following object is masked from 'package:purrr':  
##   
## lift

library(rpart.plot)  
library(pROC)

## Type 'citation("pROC")' for a citation.  
##   
## Attaching package: 'pROC'  
##   
## The following objects are masked from 'package:stats':  
##   
## cov, smooth, var

library(NHANES)

# Part 0: Data Preprocessing

## Data Import and Cleaning

We will begin by importing the NHANES 1994-2004 data. Next, we will use the select function to include the following 11 features in the tidied data set: \* Age \* Race1 \* Education \* HH\_Income \* Weight \* Height \* Pulse \* Diabetes \* BMI \* Phys\_Active \* Smoke100

Finally, we will clean the data by first applying the clean\_names function, and will remove entries with NA using na.omit.

data("NHANES")   
  
nhanes = NHANES %>%   
 select("Age", "Race1", "Education", "HHIncome", "Weight", "Height", "Pulse", "Diabetes", "BMI", "PhysActive", "Smoke100") %>%   
 janitor::clean\_names() %>%   
 na.omit()

## Centering and Scaling

Below, we center and scale these data. In general, it is always good practice to do so!

nhanes\_numeric = nhanes %>%   
 select(where(is.numeric))   
  
preprocess\_setup <- preProcess(nhanes\_numeric, method = c("center", "scale"))  
transformed.vals = predict(preprocess\_setup, nhanes\_numeric)

## Partitioning Data

For the purposes of this analysis, we will partition the data into training and testing using a 70/30 split. This process involves applying the createDataPartition function to generate a set of training and testing data with equal proportion of individual with the outcome of interest, i.e., Diabetes. The new object train\_index contains all the indexes of the rows in the original data set contained in the 70% split. The rows indexed to be in the 70% is assigned to a new training data set, and the remaining 30% is assigned to a new testing data set.

train\_index = createDataPartition(nhanes$diabetes, p = 0.7, list = FALSE)  
  
nhanes\_train <- nhanes[train\_index,]  
nhanes\_test <- nhanes[-train\_index,]

# Part I: Creating Three Different Models

For the purposes of this analysis, we will create and compare the following models:

1. Classification Tree Model
2. Support Vector Classifier Model
3. Logistic Regression Model

## 1.1 Model 1: Classification Tree Model

In the code chunk below, we will use the trainControl function to set our validation method. For the purposes of this analysis, we will use the 10-fold cross validation method.

train\_control\_tree = trainControl(method = "cv", number = 10, sampling = "down")

Next, we will create a sequence of cp parameters to try, and train the model. We will generate an accuracy metric and confusion matrix from model training.

set.seed(123)  
  
grid = expand.grid(cp = seq(0.001, 0.3, by = 0.01))  
tree\_diabetes = train(diabetes ~ .,   
 data = nhanes\_train,   
 method = "rpart",   
 trControl = train\_control\_tree,   
 tuneGrid = grid)  
  
tree\_diabetes$bestTune

## cp  
## 1 0.001

confusionMatrix(tree\_diabetes)

## Cross-Validated (10 fold) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction No Yes  
## No 63.4 2.9  
## Yes 26.2 7.4  
##   
## Accuracy (average) : 0.7085

Based on the output above, the average accuracy of Classification Tree model is **0.7198**, and the cp value is **0.001**.

## 1.2 Model 2: Support Vector Classifier Model

In the code chunk below, we will use the trainControl function to set our validation method. For the purposes of this analysis, we will use the 10-fold cross validation method and will generate predicted probabilities.

train\_control\_svm = trainControl(method = "cv", number = 10, classProbs = T)

Next, we will incorporate different values for cost (C) into the model. We will also show information about the final model, and generate the metrics of accuracy from training using the confusionMatrix function.

set.seed(123)  
  
svm\_diabetes = train(diabetes ~ .,   
 data = nhanes\_train,   
 method = "svmLinear",   
 trControl = train\_control\_svm,   
 preProcess = c("center", "scale"),   
 tuneGrid = expand.grid(C = seq(0.001, 2, length = 30)))

## maximum number of iterations reached 0.00542862 0.005106481maximum number of iterations reached 0.01141271 0.01008856maximum number of iterations reached 0.008997321 0.008195446maximum number of iterations reached 0.01000084 0.00908155maximum number of iterations reached 0.0121327 0.01077194maximum number of iterations reached 0.01094316 0.009723276maximum number of iterations reached 0.006924486 0.006498325maximum number of iterations reached 0.01010349 0.009230698maximum number of iterations reached 0.01151164 0.01014757maximum number of iterations reached 0.008651093 0.007831655maximum number of iterations reached 0.007966343 0.007423293maximum number of iterations reached 0.009041317 0.008198776maximum number of iterations reached 0.01054306 0.009400781maximum number of iterations reached 0.008361644 0.0076016maximum number of iterations reached 0.009211781 0.008399513maximum number of iterations reached 0.009223652 0.008244143maximum number of iterations reached 0.01133919 0.01011627maximum number of iterations reached 0.01131634 0.01014484maximum number of iterations reached 0.009557424 0.008665232maximum number of iterations reached 0.01209677 0.01067845maximum number of iterations reached 0.009433713 0.008481776maximum number of iterations reached 0.009035336 0.008313599maximum number of iterations reached 0.007816038 0.007226345maximum number of iterations reached 0.009041457 0.008332699maximum number of iterations reached 0.008354065 0.007714555maximum number of iterations reached 0.009306295 0.008436116maximum number of iterations reached 0.00954188 0.008623372maximum number of iterations reached 0.01031111 0.009414918maximum number of iterations reached 0.01055952 0.009640628maximum number of iterations reached 0.01240212 0.01123132maximum number of iterations reached 0.01172121 0.01019054maximum number of iterations reached 0.00980414 0.008742703maximum number of iterations reached 0.008112326 0.007448967maximum number of iterations reached 0.009492295 0.00860617maximum number of iterations reached 0.008032373 0.00741547maximum number of iterations reached 0.009125956 0.008188041maximum number of iterations reached 0.006735032 0.006288157maximum number of iterations reached 0.007104913 0.006616114maximum number of iterations reached 0.008465499 0.007645738maximum number of iterations reached 0.005835166 0.005412302maximum number of iterations reached 0.008858876 0.008147949maximum number of iterations reached 0.007139767 0.006652782maximum number of iterations reached 0.006483776 0.006170412maximum number of iterations reached 0.01080087 0.009705158maximum number of iterations reached 0.008734557 0.007964872maximum number of iterations reached 0.007332922 0.006824513maximum number of iterations reached 0.008792625 0.008192481maximum number of iterations reached 0.008608885 0.007826548maximum number of iterations reached 0.0114445 0.01009425maximum number of iterations reached 0.007858869 0.007263112maximum number of iterations reached 0.009426306 0.00864479maximum number of iterations reached 0.008288625 0.007692504maximum number of iterations reached 0.009271709 0.008625746maximum number of iterations reached 0.006472021 0.006109027maximum number of iterations reached 0.008533734 0.007914901maximum number of iterations reached 0.007925194 0.007380137maximum number of iterations reached 0.005108628 0.004876994maximum number of iterations reached 0.009022055 0.008280787maximum number of iterations reached 0.00998146 0.009092687maximum number of iterations reached 0.008869396 0.008193458maximum number of iterations reached 0.0113835 0.009930474maximum number of iterations reached 0.01058117 0.009624782maximum number of iterations reached 0.004086065 0.003931194maximum number of iterations reached 0.007083964 0.006678736maximum number of iterations reached 0.007901108 0.007382921maximum number of iterations reached 0.008263489 0.007561079maximum number of iterations reached 0.001789946 0.001764345maximum number of iterations reached 0.007560328 0.00704373maximum number of iterations reached 0.01004992 0.009245154maximum number of iterations reached 0.005158911 0.004958911maximum number of iterations reached 0.008463093 0.007675602maximum number of iterations reached 0.009129909 0.008398776maximum number of iterations reached 0.009289544 0.008551254maximum number of iterations reached 0.008498638 0.007823412maximum number of iterations reached 0.008497809 0.007842639maximum number of iterations reached 0.003450403 0.003357663maximum number of iterations reached 0.008163734 0.007555933maximum number of iterations reached 0.008310987 0.00776142maximum number of iterations reached 0.007929312 0.007383566maximum number of iterations reached 0.005895619 0.005599207maximum number of iterations reached 0.006698237 0.0063626maximum number of iterations reached 0.008245596 0.007578691maximum number of iterations reached 0.007562321 0.007202638maximum number of iterations reached 0.006857692 0.006463165maximum number of iterations reached 0.004992266 0.004840035maximum number of iterations reached 0.007810862 0.007282821maximum number of iterations reached 0.00439084 0.004249033maximum number of iterations reached 0.007244653 0.006808937maximum number of iterations reached 0.008991291 0.008309914maximum number of iterations reached 0.007676495 0.007296125maximum number of iterations reached 0.01182131 0.01016803maximum number of iterations reached 0.008421532 0.007764163maximum number of iterations reached 0.01089817 0.009634916maximum number of iterations reached 0.008543571 0.007865754maximum number of iterations reached 0.009238811 0.008273356maximum number of iterations reached 0.008570244 0.007800336maximum number of iterations reached 0.008915768 0.008170537maximum number of iterations reached 0.008856358 0.008038972maximum number of iterations reached 0.009161933 0.008314152maximum number of iterations reached 0.009737827 0.008795916maximum number of iterations reached 0.01017681 0.009327451maximum number of iterations reached 0.01063107 0.009731474maximum number of iterations reached 0.01099879 0.009751828maximum number of iterations reached 0.008124341 0.007662203maximum number of iterations reached 0.007895084 0.007287127maximum number of iterations reached 0.008867009 0.008158086maximum number of iterations reached 0.008561211 0.007954657maximum number of iterations reached 0.008113212 0.007662941maximum number of iterations reached 0.01123294 0.009919505maximum number of iterations reached 0.0061974 0.005841387maximum number of iterations reached 0.01062027 0.009485915maximum number of iterations reached 0.005669741 0.005424886maximum number of iterations reached 0.01002992 0.009181514maximum number of iterations reached 0.006756756 0.006318234maximum number of iterations reached 0.007862323 0.007245727maximum number of iterations reached 0.008328179 0.007756982maximum number of iterations reached 0.008956406 0.008345047maximum number of iterations reached 0.006399627 0.006039674maximum number of iterations reached 0.009263146 0.008495546maximum number of iterations reached 0.007588483 0.007094748maximum number of iterations reached 0.01155875 0.0103528maximum number of iterations reached 0.01275423 0.01105016maximum number of iterations reached 0.007608435 0.007052547maximum number of iterations reached 0.004871915 0.004633297maximum number of iterations reached 0.01181187 0.01066753maximum number of iterations reached 0.008740873 0.008004637maximum number of iterations reached 0.009282389 0.008473632maximum number of iterations reached 0.00823637 0.007688172maximum number of iterations reached 0.01021642 0.009204983maximum number of iterations reached 0.009535604 0.008635055maximum number of iterations reached 0.009788663 0.008917242maximum number of iterations reached 0.01303927 0.01149753maximum number of iterations reached 0.009519106 0.008609303maximum number of iterations reached 0.00581326 0.005503548maximum number of iterations reached 0.007734509 0.007179017maximum number of iterations reached 0.008811293 0.008132756maximum number of iterations reached 0.009789664 0.008883226maximum number of iterations reached 0.01092626 0.00971896maximum number of iterations reached 0.008326346 0.007731652maximum number of iterations reached 0.01043432 0.009458021maximum number of iterations reached 0.009286184 0.008482746maximum number of iterations reached 0.007070982 0.006711745maximum number of iterations reached 0.01186965 0.01047994maximum number of iterations reached 0.007043815 0.006603133maximum number of iterations reached 0.01036082 0.009383934maximum number of iterations reached 0.007686615 0.007018863maximum number of iterations reached 0.009130037 0.008384841maximum number of iterations reached 0.009100675 0.008362973maximum number of iterations reached 0.0103734 0.009455947maximum number of iterations reached 0.009617324 0.008652676maximum number of iterations reached 0.01102178 0.01000954maximum number of iterations reached 0.01020928 0.009221936maximum number of iterations reached 0.00949869 0.008585488maximum number of iterations reached 0.00449828 0.004316614maximum number of iterations reached 0.01034589 0.00937329maximum number of iterations reached 0.008641661 0.008031554maximum number of iterations reached 0.01176587 0.01033569maximum number of iterations reached 0.01153307 0.01033287maximum number of iterations reached 0.01116551 0.01017136maximum number of iterations reached 0.009677875 0.00877176maximum number of iterations reached 0.005496159 0.005262469maximum number of iterations reached 0.01004962 0.009167051maximum number of iterations reached 0.008832541 0.008132404maximum number of iterations reached 0.01033981 0.009369818maximum number of iterations reached 0.00721042 0.006787354maximum number of iterations reached 0.007643406 0.007122625maximum number of iterations reached 0.006139999 0.005700804maximum number of iterations reached 0.008274671 0.007519708maximum number of iterations reached 0.00985009 0.008958395maximum number of iterations reached 0.00905876 0.008414721maximum number of iterations reached 0.01097163 0.00987156maximum number of iterations reached 0.004296035 0.00413715maximum number of iterations reached 0.007859622 0.00735407maximum number of iterations reached 0.008000369 0.007512615maximum number of iterations reached 0.007955069 0.007410798maximum number of iterations reached 0.01000601 0.009104758maximum number of iterations reached 0.01044468 0.009326001maximum number of iterations reached 0.01050809 0.009372982maximum number of iterations reached 0.006892915 0.006413417maximum number of iterations reached 0.01098902 0.0100282maximum number of iterations reached 0.01029479 0.009102729maximum number of iterations reached 0.01047008 0.009198343maximum number of iterations reached 0.009834617 0.008763523maximum number of iterations reached 0.009972045 0.00882866maximum number of iterations reached 0.009971299 0.008909014maximum number of iterations reached 0.009671231 0.008656705maximum number of iterations reached 0.0100707 0.009056336maximum number of iterations reached 0.008430743 0.007687005maximum number of iterations reached 0.0102894 0.009260506maximum number of iterations reached 0.01217517 0.01088553maximum number of iterations reached 0.009962228 0.008836398maximum number of iterations reached 0.00680344 0.006316021maximum number of iterations reached 0.01278016 0.01109753maximum number of iterations reached 0.008104096 0.007442913maximum number of iterations reached 0.008647067 0.007887134maximum number of iterations reached 0.01181598 0.0104694maximum number of iterations reached 0.01109575 0.009780754maximum number of iterations reached 0.01043224 0.009416131maximum number of iterations reached 0.01245156 0.01090975maximum number of iterations reached 0.008562925 0.007965511maximum number of iterations reached 0.01102547 0.009973786maximum number of iterations reached 0.01283672 0.01141271maximum number of iterations reached 0.009124534 0.008356426maximum number of iterations reached 0.008630521 0.007922728maximum number of iterations reached 0.01068058 0.009697154maximum number of iterations reached 0.009556095 0.00861782maximum number of iterations reached 0.009817918 0.009000667maximum number of iterations reached 0.01143592 0.01030521maximum number of iterations reached 0.01079842 0.009590161maximum number of iterations reached 0.0116557 0.01037619maximum number of iterations reached 0.009268855 0.008340917maximum number of iterations reached 0.008966515 0.008159544maximum number of iterations reached 0.009013578 0.008275408maximum number of iterations reached 0.009327736 0.008476545maximum number of iterations reached 0.007328304 0.006704785maximum number of iterations reached 0.009377976 0.008578087maximum number of iterations reached 0.005717854 0.005374566maximum number of iterations reached 0.008979622 0.008151044maximum number of iterations reached 0.01046611 0.009310583maximum number of iterations reached 0.009868655 0.008861758maximum number of iterations reached 0.007813143 0.0073527maximum number of iterations reached 0.008428264 0.007787186maximum number of iterations reached 0.00850764 0.007786464maximum number of iterations reached 0.008218703 0.007605805maximum number of iterations reached 0.01009493 0.009178931maximum number of iterations reached 0.008913623 0.00829576maximum number of iterations reached 0.01019639 0.009190207maximum number of iterations reached 0.006927048 0.006390068maximum number of iterations reached 0.008607113 0.00784708maximum number of iterations reached 0.01184078 0.01046785maximum number of iterations reached 0.006943424 0.006460218maximum number of iterations reached 0.009991145 0.009169467maximum number of iterations reached 0.0100874 0.00906157maximum number of iterations reached 0.004737968 0.004545258maximum number of iterations reached 0.009860459 0.008892749maximum number of iterations reached 0.009322735 0.008405645maximum number of iterations reached 0.009531121 0.008736772maximum number of iterations reached 0.007421713 0.006858752maximum number of iterations reached 0.008998586 0.008310613maximum number of iterations reached 0.009699475 0.008646206maximum number of iterations reached 0.00980173 0.008830545maximum number of iterations reached 0.01076909 0.009596841maximum number of iterations reached 0.01209906 0.01068531maximum number of iterations reached 0.009171578 0.008366614maximum number of iterations reached 0.01096984 0.009810526maximum number of iterations reached 0.008332583 0.007659831maximum number of iterations reached 0.01152098 0.01010852maximum number of iterations reached 0.009697328 0.008782058maximum number of iterations reached 0.009678582 0.008827911maximum number of iterations reached 0.009925819 0.009059868maximum number of iterations reached 0.008636266 0.008041798maximum number of iterations reached 0.007340007 0.006741384maximum number of iterations reached 0.009182104 0.008282426maximum number of iterations reached 0.01187381 0.01050167maximum number of iterations reached 0.01190998 0.01045964maximum number of iterations reached 0.008306266 0.007690449maximum number of iterations reached 0.009200073 0.008432875maximum number of iterations reached 0.00807728 0.007442158maximum number of iterations reached 0.007851825 0.007196797maximum number of iterations reached 0.008838155 0.008042746maximum number of iterations reached 0.01066778 0.009697204maximum number of iterations reached 0.01044798 0.009332175maximum number of iterations reached 0.007287072 0.006785785maximum number of iterations reached 0.009451935 0.008476079maximum number of iterations reached 0.008949336 0.008106996maximum number of iterations reached 0.0105101 0.009529236maximum number of iterations reached 0.009447646 0.008648641maximum number of iterations reached 0.008774916 0.007948311maximum number of iterations reached 0.008491261 0.007746419maximum number of iterations reached 0.007651369 0.007179874maximum number of iterations reached 0.01032671 0.0093888maximum number of iterations reached 0.006569359 0.006166233maximum number of iterations reached 0.007588797 0.007076499maximum number of iterations reached 0.008556141 0.007799814maximum number of iterations reached 0.009013436 0.008281654maximum number of iterations reached 0.009665431 0.008738233maximum number of iterations reached 0.008143944 0.007589136maximum number of iterations reached 0.00780818 0.007257858maximum number of iterations reached 0.00979646 0.008705877maximum number of iterations reached 0.006380245 0.006031916maximum number of iterations reached 0.007920093 0.007494663maximum number of iterations reached 0.007551275 0.007122439maximum number of iterations reached 0.008834711 0.008022918maximum number of iterations reached 0.007384245 0.006878332maximum number of iterations reached 0.009498357 0.008746564maximum number of iterations reached 0.00858783 0.007870901maximum number of iterations reached 0.006368656 0.006066884maximum number of iterations reached 0.005448129 0.005192895maximum number of iterations reached 0.01084916 0.009821948maximum number of iterations reached 0.007418439 0.006988384maximum number of iterations reached 0.006401381 0.006031775maximum number of iterations reached 0.007058815 0.006697368maximum number of iterations reached 0.009613917 0.008654951maximum number of iterations reached 0.007241865 0.006801721maximum number of iterations reached 0.007129485 0.006786837maximum number of iterations reached 0.006827289 0.006464889maximum number of iterations reached 0.005846477 0.005598454maximum number of iterations reached 0.006389119 0.006027724maximum number of iterations reached 0.008247334 0.007723896maximum number of iterations reached 0.006695855 0.006371998maximum number of iterations reached 0.009207291 0.008270455

svm\_diabetes$finalModel

## Support Vector Machine object of class "ksvm"   
##   
## SV type: C-svc (classification)   
## parameter : cost C = 0.001   
##   
## Linear (vanilla) kernel function.   
##   
## Number of Support Vectors : 975   
##   
## Objective Function Value : -0.924   
## Training error : 0.10382   
## Probability model included.

confusionMatrix(svm\_diabetes)

## Cross-Validated (10 fold) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction No Yes  
## No 89.6 10.4  
## Yes 0.0 0.0  
##   
## Accuracy (average) : 0.8962

Based on the output above, the accuracy of the SVC model is **0.8962**, and the cost value is 0.001.

## 1.3 Model 3: Logistic Regression Model

We will employ a similar approach as seen in Parts 1.1 and 1.2 to generate a logistic regression model. First, we will use the trainControl function to set our validation method. For the purposes of this analysis, we will use the 10-fold cross validation method.

train\_control\_logistic = trainControl(method = "cv", number = 10)

Next, we will train the algorithm by specifying model = "glm".

set.seed(123)  
  
logistic\_diabetes = train(diabetes ~ .,   
 data = nhanes\_train,   
 method = "glm",   
 trControl = train\_control\_logistic,   
 preProcess = c("center", "scale"))  
  
logistic\_diabetes$finalModel

##   
## Call: NULL  
##   
## Coefficients:  
## (Intercept) age   
## -2.790641 1.069109   
## race1Hispanic race1Mexican   
## -0.023481 -0.006227   
## race1White race1Other   
## -0.368167 0.097476   
## `education9 - 11th Grade` `educationHigh School`   
## -0.189134 -0.143887   
## `educationSome College` `educationCollege Grad`   
## -0.144553 -0.185800   
## `hh\_income 5000-9999` `hh\_income10000-14999`   
## 0.052109 0.072010   
## `hh\_income15000-19999` `hh\_income20000-24999`   
## -0.003476 0.036607   
## `hh\_income25000-34999` `hh\_income35000-44999`   
## 0.013142 0.055059   
## `hh\_income45000-54999` `hh\_income55000-64999`   
## -0.030889 0.035983   
## `hh\_income65000-74999` `hh\_income75000-99999`   
## 0.020916 0.014338   
## `hh\_incomemore 99999` weight   
## -0.122902 0.053960   
## height pulse   
## 0.132908 0.115737   
## bmi phys\_activeYes   
## 0.605368 -0.092298   
## smoke100Yes   
## 0.109753   
##   
## Degrees of Freedom: 4449 Total (i.e. Null); 4423 Residual  
## Null Deviance: 2967   
## Residual Deviance: 2382 AIC: 2436

confusionMatrix(logistic\_diabetes)

## Cross-Validated (10 fold) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction No Yes  
## No 88.8 9.4  
## Yes 0.8 1.0  
##   
## Accuracy (average) : 0.8978

Based on the output above, the accuracy of the Logistic Regression model is **0.896**.

# Part II: Comparing Three Different Models

In Part I, we generated evaluation metrics for each of the Classification Tree, SVM and Logistic Regression models. Based on the outputs generated, we know that each model had the following accuracy levels from the test set:

* Classification Tree Model accuracy: 0.7198
* SVC Model accuracy: 0.8962
* Logistic Regression Model accuracy: 0.8960

We can assess the for the most optimal model by comparing these accuracy levels. Since the SVC Model generated the highest accuracy level at 0.8692, we will proceed with the SVC model.

# Part III: Calculate Final Evaluation Metrics in Test Set with the Optimal Model

It was determined in Part II that the most optimal model is the SVC model, which generated the highest accuracy level. We will apply this model to the testing data set, and will generate evaluation metrics using confusionMatrix.

set.seed(123)  
  
svm\_pred\_diabetes\_test = predict(svm\_diabetes, nhanes\_test)  
  
confusionMatrix(svm\_pred\_diabetes\_test, nhanes\_test$diabetes)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 1709 197  
## Yes 0 0  
##   
## Accuracy : 0.8966   
## 95% CI : (0.8821, 0.91)  
## No Information Rate : 0.8966   
## P-Value [Acc > NIR] : 0.519   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 1.0000   
## Specificity : 0.0000   
## Pos Pred Value : 0.8966   
## Neg Pred Value : NaN   
## Prevalence : 0.8966   
## Detection Rate : 0.8966   
## Detection Prevalence : 1.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : No   
##

The kappa value is 0, and the Mcnemar’s Test p-value is <2e-16. The accuracy level of the SVC model is 0.8966, with a 95% confidence interval of 0.8821 to 0.91. The sensitivity of this model is 1.000 and the specificity of this model is 0.000. The reported prevalence of 0.8966.

# Part IV: Limitations Discussion

One main limitation of the SVC model is the limited interpretability of results, compared to, for example, regularized regressions. Although it is possible to generate importance factors from these models, it is not as straightforward.

A second limitation of the SVC model in practice is that it may lead to increased misclassification of points in the event where there is heavy overlap between the data points. More specifically, an optimal model may have an increased C value to better classify *training* observations; however, this same model may lead to increased misclassification of testing observations, thus causing it to yield sub-optimal testing evaluation metrics. As the SVC model allows for some level of misclassification, there is an important balance that must be struck in order to apply this model successfully in practice.