Predicting the risk of diabetes using neural networks

STATS/CSE 790 Assignment 1 (2024-01-25)

Pao Zhu Vivian Hsu (400547994)

Introduction

According to the World Health Organization (2023), diabetes is attributed to 1.5 million deaths annually and has been drastically increasing over the last 30 years (2023). As a result, early diagnosis has become critical to improve the health outcomes of diabetes patients worldwide. In this report, we utilize open source data and neural network algorithms to predict diabetes risk.

The data used in this study comes from an open source website called Kaggle (Islam et al., 2020; Larxel, 2024). It contains 520 rows of patient data collected from a hospital in Bangladesh and 17 variables including an indicator for positive or negative diabetes risk, age, and binary variables denoting the presence or absence of common diabetes symptoms (Larxel, 2024).

Methods

To begin, we first visualized the data in a bar plot as shown in Figure 1. The data set had an imbalance in the classes where 38% were negative and 62% were positive cases of risk. With this imbalance in mind, we then split the data into training and testing sets. We used random sampling to select the observations in each set and maintained the same class imbalance.

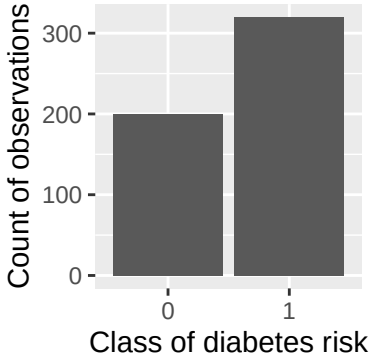

Figure 1: Boxplot of class

Next, we built three neural networks, each with a single hidden layer. A neural network is a machine learning algorithm which inputs predictors variables, computes activations in the hidden layer, and outputs a linear model that uses the activation to predict the outcome (King-Yu, 2024).

The first neural network had a size of 10 units in the hidden layer and a weight decay of 2. To improve the accuracy of the model, the second model used cross-validation to tune the size and decay parameters. As shown in Figure 2, we tested hidden layer sizes of 1 to 20 and weight decays of 1 to 5. The optimal size and decay values were 9 and 0 respectively. The last model uses bootstrap to tune the size and decay parameters. Similar to the previous model, we tested hidden layer sizes of 1 to 20 and weight decays of 1 to 5 as illustrated in Figure 2. The optimal size and decay values were 16 and 1 respectively.

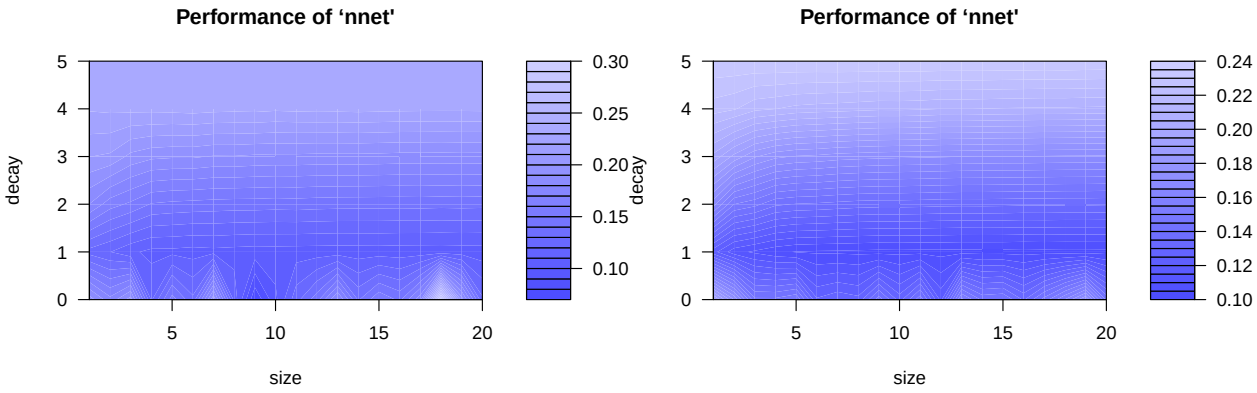

Figure 2: Decay vs. size plot for model with cross-validation (left) and bootstrap (right)

Results

Table 1 below summarizes the results of the neural networks. The accuracy for all of the models are quite high. This value is highest for the neural network that uses cross-validation, followed by the initial model, and finally the bootstrap model. Likewise, the Rand index follows a similar pattern and indicates that the most accurate models are the cross-validation model, followed by the initial model, and finally the bootstrap model. This pattern is also evident in the adjusted Rand index, which corrects for chance. Overall, these performance measures indicate that the neural network that uses cross-validation performs the best.

Table 1: Neural netowrk performance comparison

Model	Accuracy	Kappa	Rand Index	Adjusted Rand Index
Initial	93.46%	86.32%	87.73%	75.42%
Cross-validation	96.54%	92.73%	93.29%	86.56%
Bootstrap	92.69%	84.76%	86.4%	72.76%

Conclusion

Based on the results of our study, we conclude that neural networks are a strong machine learning technique to predict diabetes risk. We achieved the highest accuracy with our cross-validation model which suggests that using this model in clinical settings could aid in early detection of diabetes with an approximately 86.56% accuracy. While this is quite high, future work should be done to reduce the chance of incorrect detection and diagnosis. For example, the dispersion parameter can be tuned to improve model accuracy.

References

Islam, M. M. F., Ferdousi, R., Rahman, S., & Bushra, H. Y. (2020). Likelihood prediction of diabetes at early stage using data mining techniques. In M. Gupta, D. Konar, S. Bhattacharyya, & S. Biswas (Eds.), *Computer vision and machine intelligence in medical image analysis* (pp. 113–125). Springer Singapore. https://doi.org/10.1007/978-981-13-8798-2_12

King-Yu, S. (2024). *Statistical learning lecture 3: Neural networks*.

Larxel. (2024). *Early classification of diabetes*. https://www.kaggle.com/datasets/andrewmvd/early-diabetes-classification

World Health Organization. (2023). *Diabetes*. https://www.who.int/health-topics/diabetes#tab=tab_1

Appendix

```
# ----- PACKAGE & DATA SETUP ----- #
# Packages
rm(list=ls())
library(knitr)
library(tidyverse)
library(ggplot2)
library(nnet)
library(e1071)

# Preliminary data transformation
diabs_raw <- read.csv("diabetes_data.csv", sep=";")
diabs_int <- diabs_raw %>%
  mutate(gender = as.integer(ifelse(gender=="Male",1, ifelse(gender=="Female",0,
                             NA)))) %>%
  arrange(class)

# Helper function
as.percent <- function(value){paste0(round(value*100,2),"%")}

# ----- DATA EXPLORATION ----- #
# Check for missing data (nulls and blanks)
sum(sapply(diabs_int, function(col){ifelse(is.na(col), 1, 0)}))
sum(sapply(diabs_int, function(col){ifelse(sum(col == ""), 1, 0)}))

# Bar chart to check for class imbalance
ggplot(diabs_int %>% mutate(class=as.factor(class)),
       aes(x = class)) + geom_bar() +
  xlab("Class of diabetes risk") +
  ylab("Count of observations")
```

```r
# ----- TRAIN & TEST SETS ----- #
x <- scale(diabs_int[,-17]) # variables
cls<-class.ind(diabs_int[,17]) # class

nrow_0 <- diabs_int %>% filter(class == 0) %>% nrow() # num class 0 rows
perc_0 <- nrow_0/nrow(diabs_int) # % of class 0 rows
size_0 <- nrow(diabs_int)/2*perc_0 # num class 0 rows to sample
size_1 <- nrow(diabs_int)/2*(1-perc_0) # num class 1 rows to sample

set.seed(2024)
train <- c(sample(1:nrow_0, size_0), sample((nrow_0+1):nrow(diabs_int), size_1))

# ----- NEURAL NETWORKS ----- #
# MODEL 1: INITIAL
set.seed(2024)
nnDiabs_mod1 <- nnet(x[train,], cls[train,], size=10, decay=2, softmax=TRUE)

nnDiabs_pred1 <- predict(nnDiabs_mod1, x[-train,], type="class")
nnDiabs_tab1 <- table(diabs_int[-train,17], nnDiabs_pred1)

# MODEL 2: CROSS-VALIDATION
# Optimize size and decay rate using cross-validation
set.seed(2024)
nnDiabs_cv = tune.nnet(class~., data = diabs_int[train,], size = 1:20, decay=0:5,
                       tunecontrol = tune.control(sampling = "cross", cross=5))
plot(nnDiabs_cv)
set.seed(2024)
nnDiabs_mod2<-nnet(x[train,], cls[train,], size=9, decay=0, softmax=TRUE)
nnDiabs_pred2<-predict(nnDiabs_mod2, x[-train,], type="class")
nnDiabs_tab2<-table(diabs_int[-train,17],nnDiabs_pred2)
```

```r
# MODEL 3: BOOTSTRAP SAMPLING
# Optimize size and decay rate using bootstrap sampling
set.seed(2024)
nnDiabs_boot = tune.nnet(class~., data = diabs_int[train,], size = 1:20, decay=0:5,
                         tunecontrol = tune.control(sampling = "boot", nboot=20))
plot(nnDiabs_boot)
set.seed(2024)
nnDiabs_mod3 <- nnet(x[train,], cls[train,], size=16, decay=1, softmax=TRUE)
nnDiabs_pred3 <- predict(nnDiabs_mod3, x[-train,], type="class")
nnDiabs_tab3 <- table(diabs_int[-train,17], nnDiabs_pred3)

# RESULTS
model <- c("Initial", "Cross-validation", "Bootstrap")
accuracy <- c(classAgreement(nnDiabs_tab1)$diag,
              classAgreement(nnDiabs_tab2)$diag,
              classAgreement(nnDiabs_tab3)$diag)
kappa <- c(classAgreement(nnDiabs_tab1)$kappa,
           classAgreement(nnDiabs_tab2)$kappa,
           classAgreement(nnDiabs_tab3)$kappa)
rand <- c(classAgreement(nnDiabs_tab1)$rand,
          classAgreement(nnDiabs_tab2)$rand,
          classAgreement(nnDiabs_tab3)$rand)
crand <- c(classAgreement(nnDiabs_tab1)$crand,
           classAgreement(nnDiabs_tab2)$crand,
           classAgreement(nnDiabs_tab3)$crand)
nnDiabs_summary <- data.frame("Model" = model,
                              "Accuracy" = as.percent(accuracy),
                              "Kappa" = as.percent(kappa),
                              "Rand Index" = as.percent(rand),
                              "Adjusted Rand Index" = as.percent(crand),
                              check.names = FALSE)
```

```
kable(nnDiabs_summary)
```