MULTINOMIAL LOGISTIC REGRESSION

evenif

8/16/2021

library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5 v purrr 0.3.4  
## v tibble 3.1.2 v dplyr 1.0.6  
## v tidyr 1.1.3 v stringr 1.4.0  
## v readr 1.4.0 v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(gtsummary)  
library(nnet)

# Loading the library  
library(rattle.data)  
# Loading the wine data  
data(wine)

# Checking the structure of wine dataset  
str(wine)

## 'data.frame': 178 obs. of 14 variables:  
## $ Type : Factor w/ 3 levels "1","2","3": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Alcohol : num 14.2 13.2 13.2 14.4 13.2 ...  
## $ Malic : num 1.71 1.78 2.36 1.95 2.59 1.76 1.87 2.15 1.64 1.35 ...  
## $ Ash : num 2.43 2.14 2.67 2.5 2.87 2.45 2.45 2.61 2.17 2.27 ...  
## $ Alcalinity : num 15.6 11.2 18.6 16.8 21 15.2 14.6 17.6 14 16 ...  
## $ Magnesium : int 127 100 101 113 118 112 96 121 97 98 ...  
## $ Phenols : num 2.8 2.65 2.8 3.85 2.8 3.27 2.5 2.6 2.8 2.98 ...  
## $ Flavanoids : num 3.06 2.76 3.24 3.49 2.69 3.39 2.52 2.51 2.98 3.15 ...  
## $ Nonflavanoids : num 0.28 0.26 0.3 0.24 0.39 0.34 0.3 0.31 0.29 0.22 ...  
## $ Proanthocyanins: num 2.29 1.28 2.81 2.18 1.82 1.97 1.98 1.25 1.98 1.85 ...  
## $ Color : num 5.64 4.38 5.68 7.8 4.32 6.75 5.25 5.05 5.2 7.22 ...  
## $ Hue : num 1.04 1.05 1.03 0.86 1.04 1.05 1.02 1.06 1.08 1.01 ...  
## $ Dilution : num 3.92 3.4 3.17 3.45 2.93 2.85 3.58 3.58 2.85 3.55 ...  
## $ Proline : int 1065 1050 1185 1480 735 1450 1290 1295 1045 1045 ...

## We now split the dataset into train and test using “sample\_frac()” function from {dplyr} package

# Using sample\_frac to create 70 - 30 slipt into test and train  
train <- sample\_frac(wine, 0.7)  
sample\_id <- as.numeric(rownames(train)) # rownames() returns character so as.numeric  
test <- wine[-sample\_id,]

## Remember when we build logistic models we need to set one of the levels of the dependent variable as a baseline. We achieve this by using relevel() function. Making “3” the reference level here

# Setting the basline   
train$Type <- relevel(train$Type, ref = "3")

## Once the baseline has been specified, we use multinom() function to fit the model and then use summary() function to explore the beta coefficients of the model.

## We shall use -1 in the formula to delete the intercept. We think that it does not make sense in the model and so we remove it.

# Training the multinomial model  
multinom.fit <- multinom(Type ~ Alcohol + Color -1, data = train)

## # weights: 9 (4 variable)  
## initial value 137.326536   
## iter 10 value 75.214712  
## final value 74.556832   
## converged

# Checking the model  
summary(multinom.fit)

## Call:  
## multinom(formula = Type ~ Alcohol + Color - 1, data = train)  
##   
## Coefficients:  
## Alcohol Color  
## 1 0.2980135 -0.6363204  
## 2 1.0287619 -2.8939478  
##   
## Std. Errors:  
## Alcohol Color  
## 1 0.08224098 0.1784825  
## 2 0.15829172 0.4761149  
##   
## Residual Deviance: 149.1137   
## AIC: 157.1137

The output of summary contains the table for coefficients and a table for standard error. Each row in the coefficient table corresponds to the model equation. The first row represents the coefficients for Type 2 wine in comparison to our baseline which is Type 3 wine and the second row represents the coefficients for Type 2 wine in comparison to our baseline which is Type 3 wine.

## This ratio of the probability of choosing Type 2 wine over the baseline that is Type 3 wine is referred to as relative risk (often described as odds). However, the output of the model is the log of odds. To get the relative risk IE odds ratio, we need to exponentiate the coefficients.

## extracting coefficients from the model and exponentiate  
exp(coef(multinom.fit))

## Alcohol Color  
## 1 1.34718 0.52923621  
## 2 2.79760 0.05535724

## We will now check the model accuracy by building classification table. So let us first build the classification table for training dataset and calculate the model accuracy

# Predicting the values for train dataset  
train$precticed <- predict(multinom.fit, newdata = train, "class")  
   
# Building classification table  
ctable <- table(train$Type, train$precticed)  
   
# Calculating accuracy - sum of diagonal elements divided by total obs  
round((sum(diag(ctable))/sum(ctable))\*100,2)

## [1] 70.4

## Accuracy in training dataset is 68.8%. We now repeat the above on the unseen dataset that tests dataset

# Predicting the values for train dataset  
test$precticed <- predict(multinom.fit, newdata = test, "class")  
  
# Building classification table  
ctable <- table(test$Type, test$precticed)  
  
# Calculating accuracy - sum of diagonal elements divided by total obs  
round((sum(diag(ctable))/sum(ctable))\*100,2)

## [1] 9.43

The accuracy of the test dataset turns out to be 18.4% less as compared to training dataset. So we have a problem of overfitting here. Despite what accuracy we get the process of building the multinomial logistic regression remains the same.

WHAT NEXT? Next, we suggest you solve this problem by achieving better results and solving the overfitting problem. To start with trying the following checks –

1. Look for multicollinearity.

If it exists – a. Remove variable or b. Run factor analysis

1. Check for outliers and do the necessary treatment.
2. Try different data transformations for independent variables.

Do share your final results in the comments below. In case you have further queries feel free to initiate the discussion. I would be happy to help you