Week 4 Practical

Logistic Regression

Overview

Logistic regression is a classification method that is almost identical to fitting a regression curve, except that the outcome is a categorical variable. In the simplest case, the outcome variable is binary, which means that the outcome is either 0 or 1.

R has a very easy and straightforward function for fitting a logistic regression model – `glm`.

Objectives

- Understand and model logistic regression.
- Interpret the obtained logistic regression model.
- Evaluate the obtained logistic regression model.

Sharing results

While tasks should be self-explanatory, some of the challenges might be more complex. Feel free to share your results or questions in the course discussion forum. Results to all the challenges will be available UPON REQUEST (discussion forum or email).

Dataset

In this practical, we will be using a dataset that was derived from one of the most popular datasets available at UCI Machine Learning Repository (https://archive.ics.uci.edu/). The original dataset has `quality` as the output variable that represents a wine quality score on the scale from 0 to 10. The derived dataset contains only records that belong to a derived `quality_class`, being labeled as 0 – medium wine quality and 1 – high quality of wine.

Table 1. Wine quality dataset description

Column Name	Description
fixed acidity	tartaric acid - g/dm3; most acids involved with wine or fixed or nonvolatile (do not evaporate readily)
volatile acidity	acetic acid - g/dm3; the amount of acetic acid in wine, which at too high of levels can lead to an unpleasant, vinegar taste
citric acid	g/dm3; found in small quantities, citric acid can add 'freshness' and flavor to wines
residual sugar	g/dm3; the amount of sugar remaining after fermentation stops, it's rare to find wines with less than 1 gram/liter and wines with greater than 45 grams/liter are considered sweet
chlorides	sodium chloride - g/dm3; the amount of salt in the wine
free sulfur dioxide	mg/dm3; the free form of SO2 exists in equilibrium between molecular SO2 (as a dissolved gas) and bisulfite ion; it prevents microbial growth and the oxidation of wine
total sulfur dioxide	mg/dm3; amount of free and bound forms of S02; in low concentrations, SO2 is mostly undetectable in wine, but at free SO2 concentrations over 50 ppm, SO2 becomes evident in the nose and taste of wine
density	g/cm3; the density of water is close to that of water depending on the percent alcohol and sugar content
pH	describes how acidic or basic a wine is on a scale from 0 (very acidic) to 14 (very basic); most wines are between 3-4 on the pH scale
sulphates	potassium sulphate - g/dm3; a wine additive which can contribute to sulfur dioxide gas (S02) levels, which acts as an antimicrobial and antioxidant
alcohol	% by volume; the percent alcohol content of the wine
quality class	0 for the medium quality wine (original scale 4-6) and 1 for the high-quality wine (original scale 7-9)

Before getting started

Before going through the tasks, make sure you have a new R Notebook created in RStudio, as shown in Figure 1.

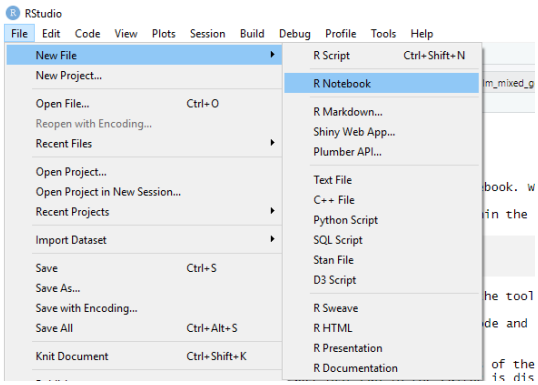

Figure 1. Opening R Notebook in RStudio

Each section of the code (marked by the grey box) should be executed within a single chunk in your Notebook.

Task 1. Fitting a logistic regression model in R

Similar to previous weeks, there are a few steps we need to perform before we can fit a model.

We will continue to use the same dataset as we have over the last several weeks. This should make it easier to compare different approaches to building a classifier.

```
# Load libraries

library(pscl)
library(ROCR)
```

```
# Load data

data <-
read.csv(url("https://raw.githubusercontent.com/sreckojoksimovic/infs5100/ma
in/wine-data.csv"))
```

There might be a problem with the column names when you load the dataset, so we will make sure the columns (features) are properly labeled. As we discussed in the lecture, it is important to scale variables before fitting a logistic regression model (please refer to the end of the video).

```
# Make sure quality_class is factor

names(data) <- c("fixed_acidity", names(data)[2:12])


data$fixed_acidity <- scale(data$fixed_acidity, scale = TRUE, center = TRUE)

data$volatile_acidity <- scale(data$volatile_acidity, scale = TRUE, center =
TRUE)

data$citric_acid <- scale(data$citric_acid, scale = TRUE, center = TRUE)

data$residual_sugar <- scale(data$residual_sugar, scale = TRUE, center =
TRUE)

data$chlorides <- scale(data$chlorides, scale = TRUE, center = TRUE)

data$free_sulfur_dioxide <- scale(data$free_sulfur_dioxide, scale = TRUE,
center = TRUE)

data$total_sulfur_dioxide <- scale(data$total_sulfur_dioxide, scale = TRUE,
center = TRUE)

data$density <- scale(data$density, scale = TRUE, center = TRUE)

data$pH <- scale(data$pH, scale = TRUE, center = TRUE)

data$sulphates <- scale(data$sulphates, scale = TRUE, center = TRUE)

data$alcohol <- scale(data$alcohol, scale = TRUE, center = TRUE)
```

As a good practice, let's summarize the obtained dataset.

```
# Data input validation

head(data)

summary(data)
```

Further, we will split data into training and test datasets.

```
# Split data into training and test datasets. We will use 70%/30% split
# again.
set.seed(123)

dat.d <- sample(1:nrow(data),size=nrow(data)*0.7,replace = FALSE) #random
selection of 70% data.

train.data <- data[dat.d,] # 70% training data

test.data <- data[-dat.d,] # remaining % test data
```

Perhaps even easier than with other models, fitting a logistic regression model is fairly straightforward.

```
model <- glm(quality_class ~., family=binomial(link='logit'),
data=train.data)
```

What happened here?

- We simply specified that our model would use `quality_class` as the outcome and all the other features as predictors - `quality_class ~.` – Instead of "." We could also specify the subset of variables we want to use – for example, `citric_acid + density`.
- `family` – this object provides a convenient way to specify the details of the models used by a function. In this case, we use binomial, as we are fitting a logistic regression model. Please refer to the glm documentation for other options.
-

Task 2. Interpreting a logistic regression model

Once we have a model, it is important to understand how to interpret the output of the model.

```
summary(model)
```

```
Call:
glm(formula = quality_class ~ ., family = binomial(link = "logit"),
    data = train.data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.9115  -0.4628  -0.2356  -0.1410   2.9099

Coefficients:
                      Estimate Std. Error z value Pr(>|z|)
(Intercept)          -2.70156    0.16058 -16.824  < 2e-16 ***
fixed_acidity         0.50867    0.26622   1.911  0.05605 .
volatile_acidity     -0.24575    0.15766  -1.559  0.11906
citric_acid           0.26652    0.19489   1.368  0.17146
residual_sugar        0.33147    0.13617   2.434  0.01492 *
chlorides            -0.52590    0.21648  -2.429  0.01513 *
free_sulfur_dioxide   0.11237    0.15292   0.735  0.46245
total_sulfur_dioxide -0.45866    0.17664  -2.597  0.00941 **
density              -0.65174    0.25497  -2.556  0.01059 *
pH                    0.02047    0.19031   0.108  0.91433
sulphates             0.60421    0.10715   5.639 1.71e-08 ***
alcohol               0.68291    0.17293   3.949 7.84e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 857.87  on 1074  degrees of freedom
Residual deviance: 609.09  on 1063  degrees of freedom
AIC: 633.09

Number of Fisher Scoring iterations: 6
```

We can see that `residual_sugar`, `sulphates` and `alcohol` are positively and significantly associated with the probability that wine is of high quality. On the other hand, `chlorides`, `total_sulfur_dioxide`, and `density` are also significantly, but negatively associated with the probability that wine is of high quality.

Please remember that in the logit model the response variable is log odds: ln(odds) = ln(p/(1-p)) = a*x1 + b*x2 + … + z*xn. This means that one unit increase in `residual_sugar`, increases the log odds by 0.33.

After understanding the output of the model summary, we should take a look at the table of deviance.

```
anova(model, test="Chisq")
```

```
Analysis of Deviance Table

Model: binomial, link: logit

Response: quality_class

Terms added sequentially (first to last)


                     Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
NULL                                  1074     857.87
fixed_acidity         1   11.399      1073     846.47 0.0007348 ***
volatile_acidity      1   67.045      1072     779.43 2.654e-16 ***
citric_acid           1    4.386      1071     775.04 0.0362378 *
residual_sugar        1    1.160      1070     773.88 0.2813884
chlorides             1   19.750      1069     754.13 8.826e-06 ***
free_sulfur_dioxide   1    3.187      1068     750.94 0.0742421 .
total_sulfur_dioxide  1   14.265      1067     736.68 0.0001588 ***
density               1   64.066      1066     672.61 1.203e-15 ***
pH                    1    6.588      1065     666.02 0.0102648 *
sulphates             1   41.208      1064     624.81 1.368e-10 ***
alcohol               1   15.723      1063     609.09 7.333e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

The difference between the `NULL` deviance and the residual deviance shows how our model is doing compare to a model with only the intercept. The wider this gap, the better model. Analyzing the output above, we can see the drop in deviance when adding each variable one at a time. All the variables, except for `residual_sugar` and `free_sulfur_dioxide`, significantly improve the model.

While no exact equivalent to the R^2 of linear regression exists, the McFadden R^2 index can be used to assess the model fit.

```
pR2(model)
```

```
fitting null model for pseudo-r2
         llh      llhNull           G2     McFadden         r2ML         r2CU
-304.5458984 -428.9345978  248.7773989    0.2899946    0.2065945    0.3757770
```

Values between 0.2-0.4 for the McFadden R^2 represent the excellent fit. Please refer to this post for further discussion https://stats.stackexchange.com/questions/82105/mcfaddens-pseudo-r2-interpretation.

Task 3. Evaluating the predictive power of the model

Finally, as we did with the previous models, we will calculate various evaluation metrics, to assess the predictive ability of our model.

```
fitted.results <- predict(model, newdata=test.data, type='response')

fitted.results <- ifelse(fitted.results > 0.5,1,0)

confusionMatrix(as.factor(fitted.results), as.factor(test.data[, 12]))
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 379  44
         1  12  26

               Accuracy : 0.8785
                 95% CI : (0.8452, 0.9069)
    No Information Rate : 0.8482
    P-Value [Acc > NIR] : 0.03698

                  Kappa : 0.4194

 Mcnemar's Test P-Value : 3.435e-05

            Sensitivity : 0.9693
            Specificity : 0.3714
         Pos Pred Value : 0.8960
         Neg Pred Value : 0.6842
             Prevalence : 0.8482
         Detection Rate : 0.8221
   Detection Prevalence : 0.9176
      Balanced Accuracy : 0.6704

       'Positive' Class : 0
```

Challenge 1. We talked in previous weeks about feature selection. Also, we implemented feature selection in the previous practical.

Can you add feature selection to the pipeline outlined in this practical?