Logistic regression

STAT30270 – Statistical Machine Learning

Contents

1 Logistic regression

The main function for fitting a logistic regression model with binary response variable is `glm`. Note that this function is used more generally to fit generalized linear models and to estimate a logistic regression we need to set the argument `family = "binomial"`.

The dataset `spam` available in package `kernlab` contains 57 features extracted from the content of emails which were classified as `spam` or `nonspam`. The first 48 variables contain the frequency of the variable name (e.g., `business`) in the email. If the variable name starts with `num` (e.g., `num650`) it indicates the frequency of the corresponding number (e.g., `650`). The variables 49-54 indicate the frequency of the special characters `;`, `(`, `[`, `!`, `\$`, and `\#`. The variables 55-57 contain the average, longest and total run-length of capital letters. The last variable, `type` indicates if an email is `spam` or `nonspam`. See `?spam` for additional information. The task is to predict if an email is `spam` or not, using the subset of features related to the frequencies of the special characters and to the statistics of the capital letters (i.e. columns 49-56 + column 57 corresponding to the target).

```
# load the data
data("spam", package = "kernlab")

set <- 49:57    # select relevant variables
data <- spam[,c(set, 58)]

str(data)    # check if all variables are coded correctly
```

In the dataset, the target variable `type` is correctly coded as factor, so there is no need to transform it. The other predictor variables are all numeric. The response variable `type`, indicates whether an email is spam or not. We fit a logistic regression where the response is function of all the other variables. The task is to predict whether an email is spam or not given the collection of features related to particular character and statistics of capital letters.

```
fit <- glm(type ~ ., data = data, family = "binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
# the '.' is short for including all the predictor variables except the response
# see '?formula' for further info

summary(fit)
```

1.1 Task

- The `glm` function returns the warning `glm.fit: fitted probabilities numerically 0 or 1 occurred`. Can you diagnose what is the cause of the warning?
- Which are the predictor variables significantly affecting the probability that an email is spam? Consider a significance level of 10^{-5}.
- Choose a few input variables and provide an interpretation of their coefficients in terms of the odds.

1.2 Predictive performance - Accuracy and ROC

The estimated probabilities can be used to perform predictions on the target variable and consequently classify the units observed on the input variables. To do so, a threshold τ is used, then if the estimated probability for observation i is greater than τ, the observation is classified as $y_i = 1$, otherwise as $y_i = 0$.

We can classify the observations for $\tau = 0.5$ running the following lines of code. Function `fitted` is employed to extract the estimated probabilities from the model (see `?fitted`).

```r
tau <- 0.5
p <- fitted(fit)
pred <- ifelse(p > tau, 1, 0)

# cross tabulation between observed and predicted
table(data$type, pred)

# compute accuracy for given tau
tab <- table(data$type, pred)
sum(diag(tab))/sum(tab)
```

Different measures for assessing the predictive performance of the logistic regression model can be computed for varying values of the discrimination threshold τ. Package `ROCR` can be used to calculate many performance measures. The package is very flexible, see the vignette `vignette("ROCR")`.

To use the functionalities of the package, we first need to create a `prediction` object, providing in input the estimated probabilities and the actual class values of the response variable.

```r
library(ROCR)
pred_obj <- prediction(fitted(fit), data$type)
```

Function `performance` then can be applied to calculate different performance measures. Have a look at the help page `?performance` for all the measures available. We can use the function to plot the ROC curve.

```r
roc <- performance(pred_obj, "tpr", "fpr")
plot(roc)
abline(0, 1, col = "darkorange2", lty = 2)   # add bisect line

# compute the area under the ROC curve
auc <- performance(pred_obj, "auc")
auc@y.values
# note that we access the value of auc in the list with '@'
# this is because the object-list is of class S4
```

The same function is employed to compute sensitivity and specificity, as well as accuracy. In relation to the ROC curve, the optimal threshold τ can be found maximizing the sum of sensitivity and specificity for different values of τ.

```
sens <- performance(pred_obj, "sens")
spec <- performance(pred_obj, "spec")

tau <- sens@x.values[[1]]
sens_spec <- sens@y.values[[1]] + spec@y.values[[1]]
best_roc <- which.max(sens_spec)
plot(tau, sens_spec, type = "l")
points(tau[best_roc], sens_spec[best_roc], pch = 19, col = adjustcolor("darkorange2", 0.5))

tau[best_roc]   # optimal tau according to the ROC curve

# classification for optimal tau
pred <- ifelse(fitted(fit) > tau[best_roc], 1, 0)
table(data$type, pred)

# accuracy for optimal tau
acc <- performance(pred_obj, "acc")
acc@y.values[[1]][best_roc]
#
# sensitivity and specificity for optimal tau
sens@y.values[[1]][best_roc]
spec@y.values[[1]][best_roc]
```

1.3 Predictive performance - Precision/Recall

Similarly to the previous section, we can use the package to produce a precision/recall curve and compute the associated area under the curve.

```
# produce precision/recall curve
pr <- performance(pred_obj, "prec", "rec")
plot(pr)

# compute area under the PR curve
aucpr <- performance(pred_obj, "aucpr")
aucpr@y.values
```

We can identify the optimal threshold τ related to the precision/recall curve and the F_1 score. In this case, the optimal value of τ is the one that maximizes the F_1 score.

```
perf_f <- performance(pred_obj, "f")

tau <- perf_f@x.values[[1]]
f <- perf_f@y.values[[1]]
best_pr <- which.max(f)
plot(tau, f, type = "l")
points(tau[best_pr], f[best_pr], pch = 19, col = adjustcolor("darkorange2", 0.5))

tau[best_pr]   # optimal tau according to the PR curve

# classification for optimal tau
pred <- ifelse(fitted(fit) > tau[best_pr], 1, 0)
```

```
table(data$type, pred)

# accuracy and F1 score for optimal tau
acc <- performance(pred_obj, "acc")
acc@y.values[[1]][best_pr]
f[best_pr]
```

1.4 Task

- In this example the classification threshold derived from the ROC curve is equal to the threshold derived from the PR curve. Can you provide an explanation of why?

2 Multinomial logistic regression

Multinomial logistic regression is an extension of the standard logistic regression model to the case where the response variable can take more than two classes. The framework is usually applied for multi-class classification problems. To fit a multinomial logistic regression we can use the function `multinom` available in the package `nnet`.

We consider an example in application to classification of vehicles. The task is to classify a given silhouette as one of four types of vehicle, using a set of numerical features extracted from the silhouette of a vehicle. The dataset is `Vehicle` and is available in the package `mlbench`. See `?Vehicle` for more details.

```
# load packages and data
library(nnet)
data("Vehicle", package = "mlbench")

# the target variable is the vehicle class
fit <- multinom(Class ~ ., data = Vehicle)

# note that the algorithm stopped without reaching convergence
# this is because it reached the maximum allowed number of iterations (100 by default)
# we fit the model again increasing the number of iterations.
fit <- multinom(Class ~ ., data = Vehicle, maxit = 300)

summary(fit)
```

Similarly to an object of class `glm`, we can use function `predict` to extract predicted quantities from the model. In particular, here function `predict` computes the estimated class labels for each observation by default. To extract the estimated probabilities, one can set the argument `type = "probs"`. Alternatively, the slot `$fitted.values` of the fitted object contains the estimated probabilities. The estimated classes can be used to compute different metrics.

```
# compare observed classes and predicted ones
tab <- table(true = Vehicle$Class, pred = predict(fit))
tab

# compute accuracy
sum( diag(tab) ) / sum(tab)
#
# compute class sensitivity
cbind( tab, c_sens = diag(tab)/rowSums(tab) )
#
# compute class false positive rate
rec <- diag(tab)/colSums(tab) # class specific recall
1 - rec
```

2.1 Task

The dataset `Vowel` in package `mlbench` records features employed to characterize the eleven steady state vowels of British English: hid, hId, hEd, hAd, hYd, had, hOd, hod, hUd, hud, hed. See `?Vowel` for further information. The data include 10 predictor variables and the target variable `Class` indicating the vowel type. Fit a multinomial regression model to predict the vowel type and assess its performance.

3 Heart disease indicators dataset

Heart-related diseases are among the most prevalent chronic diseases in the United States. Preventive identification of at-risk subjects and of the factors associated to hearth-related conditions is paramount for effective prevention of negative outcomes (like hearth attacks) and testing.

The Behavioral Risk Factor Surveillance System (BRFSS) is a health-related survey that collects state data about U.S. residents regarding their health-related risk behaviors, chronic health conditions, and use of preventive services. The dataset `data_heart_disease_BRFSS2015.csv` contains records for 253,680 respondents of the survey. For each subject, information is available on whether the subject had a heart-related disease (`HeartDiseaseorAttack`), and additional information including general behavior, demographic characteristics, self-reported health status, and disease history. More information is available at the link 'https://www.kaggle.com/alexteboul/heart-disease-health-indicators-dataset'.

We load the data and convert to the appropriate format the variables.

```
data <- read.csv("data/data_heart_disease_BRFSS2015.csv")

# make sure that binary/categorical variables are correctly encoded as factor
data[,c(1:4,6:14,18:19)] <- lapply( data[,c(1:4,6:14,18:19)], factor )
str(data)

# classes are highly imbalanced
table(data$HeartDiseaseorAttack)
table(data$HeartDiseaseorAttack)/nrow(data)
```

3.1 Task

- Fit a logistic regression model to predict the occurrence of an heart-related disease given some or all of the available covariates.
- Assess the performance of your logistic regression model. In doing so, compare the quality of the predictions (1) when the probability threshold is set to 0.5, versus (2) the quality of the predictions when the probability threshold is selected using sensitivity and specificity, and (3) when the classification threshold is selected using precision/recall.
- Which predictive performance assessment procedure and metrics are most appropriate for these data?