Logistic Regression Lab activity - Instructions

Agenda

1. Get the data
2. Data Pre-processing
3. Build a model
4. Predictions
5. Model validation

Dataset: "bank.txt"
Problem statement : Classifying whether the subject subscribes to a term deposit or not
Data Description:

The dataset is from a bank, using which we have to predict whether the subject subscribes to a term deposit or not

The dataset has the following attributes:

1 - age (numeric)

2 - job : type of job (categorical: "admin.","unknown","unemployed","management","housemaid","entrepreneur","student", "blue-collar","self-employed","retired","technician","services")

3 - marital : marital status (categorical: "married","divorced","single"; note: "divorced" means divorced or widowed)

4 - education (categorical: "unknown","secondary","primary","tertiary")

5 - default: has credit in default? (binary: "yes","no")

6 - balance: average yearly balance, in euros (numeric)

7 - housing: has housing loan? (binary: "yes","no")

8 - loan: has personal loan? (binary: "yes","no")

9 - contact: contact communication type (categorical: "unknown","telephone","cellular")

10 - day: last contact day of the month (numeric)

11 - month: last contact month of year (categorical: "jan", "feb", "mar", ..., "nov", "dec")

12 - duration: last contact duration, in seconds (numeric)

13 - campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)

14 - pdays: number of days that passed by after the client was last contacted from a previous campaign (numeric, -1 means client was not previously contacted)

15 - previous: number of contacts performed before this campaign and for this client (numeric)

16 - poutcome: outcome of the previous marketing campaign (categorical: "unknown","other","failure","success")

Response Variable (desired target):

17 - y : has the client subscribed to a term deposit?(binary: "yes","no")

Get the data(and understand it):

1. Read the bank.txt into R environment. Observe that in the earlier activities, you have read 'CSV' files. Use "read.table" command instead of "read.csv" command and save the results in a dataframe called 'bank_data' . The syntax is exactly the same as "read.csv".
2. Get the structure of the data frame and also the head of the dataframe.
3. Look at the individual variables and determine whether they are meant to be Numerical or categorical. Also see if they are assigned to correct data-types by default.
4. Use the summary command to see if you how the columns are distributed and if scaling is required or not.

Data Pre-processing:

5. Use earlier activities to see if there are any null values in each of the columns.
6. Use 'caret' package to train-test split using stratified sampling. Below is a sample code

```
library(caret)
set.seed(786)
```

```
# The argument "y" to the createDataPartition() function is
the response variable
# The argument "p" is the percentage of data that goes to
training
# The argument "list" should be input a boolean (T or F).
Remember to put list = F, else the output is going to  be a
list and your data can't be subsetted with it
train_rows <- createDataPartition(bank_data$y, p = 0.7, list =
F)
train_data <- bank_data[train_rows, ]
test_data <- bank_data[-train_rows, ]
```

Model Building

7. Apply Logistic regression on the data using ‘glm’ function. The syntax is very similar to ‘lm’ function used earlier. A sample code is given below.

```
log_reg <- glm(y~., data = train_data, family = binomial)
```

8. Perform summary function on the model object(log_reg) and see if the individual coefficients are significant.

Predictions

9. Use ‘predict’ function on the model object to get the predictions on the train dataframe. Observe the predictions and the scale. Ideally the output is expected to be in range of (0,1). But the output of Logistic regression is Log-odds. To convert the output into probability, add a parameter type = "response" in the ‘predict’ function. Sample code is given below.

```
prob_train <- predict(log_reg, type = "response")
```

10. Below is the code for ROC curve.

```
library(ROCR)
pred <- prediction(prob_train, train_data$y)
perf <- performance(pred, measure="tpr", x.measure="fpr")
plot(perf, col=rainbow(10), colorize=T,
print.cutoffs.at=seq(0,1,0.05))
```

11. FInd the AUC.

```
perf_auc <- performance(pred, measure="auc")
# Access the auc score from the performance object
auc <- perf_auc@y.values[[1]]
print(auc)
```

12. Getting final Predictions in terms of 0 or 1 by choosing a proper cut off on the predicted probability.

```
prob_test <- predict(log_reg, test_data, type = "response")
preds_test <- ifelse(prob_test > 0.5, "yes", "no")
```

Model Validation

13. Build the confusion matrix by comparing actuals and predicted values. For this, apply table function on the two

14. Get the error metrics from confusion matrix. For this, use ‘confusionMatrix’ function from ‘caret’ package.

A few questions to be investigated:

What do you think about the performance of the model?
What is the impact of choosing the cut-off on the performance? (try with different variables)
What is the best metric for this problem statement?
Will standardising the data help?