RPL Project ReadMe

Jamie Boyle

30/11/2021

Contents

1 Introduction

This document contains guidance about the data and R/Python scripts related to the author's work on the RPL project.

Note: Much of this work was written during my dissertation, whilst I was still in the middle of learning about the programming languages and all the statistical methods. There are thus many areas which could be improved, and which I would do differently if I were to start this project over again. In particular, much of the data cleaning that is done in R or Python could be better done within Excel. Additionally, the scripts for producing MCMC chains are written in such a way that makes it somewhat cumbersome to change what covariates are included. Apologies for this, and do not be afraid to make changes!

1.1 Software and Packages

To run everything outlined in this document requires Excel, R, and Python. For R and Python you will also need to install the following packages:

R Packages:

- survival - used for survival analysis
- coda - used for convergence analysis for the MCMC

Python Packages (these should all be pre-installed if you use Anaconda):

- numpy - general purpose package for working with vectors and matrices
- pandas - used for importing data from Excel
- matplotlib - used for plotting
- csv - used in MCMC convergence analysis to record chains in a spreadsheet
- random - used for data partitioning in the cross-validation scripts

1.2 Project Reports

There are two project reports that contain much more detail on the analysis carried out so far, and provide more context to the scripts discussed here. My dissertation (`Report.pdf`) contains the most detail on the methods used, however note that the analysis in this report was carried out on an older data set (version of September 2020). The write up of the work I did over the Summer (`RPLSummerWriteUp.pdf`) contains the details of when we ran the analysis on a more up to date version (version of July 2021), and also contains material on validation not in the dissertation.

All the scripts in this report relate to the July 2021 data set.

There are points of conflict between the findings of the two reports, particularly in the fits of the cure rate models. It would be of interest to fit these models to further data which I understand you are getting soon.

2 Data

2.1 Training Data

The training data is contained in the file `RFE002_demo_baseline.RCS.csv`, which contains both covariate and pregnancy outcome information.

The files `DataSetup.R` and `DataSetup.py` contain script to clean up the data and prepare it for analysis. This is probably better done within Excel, but for now these scripts must be run before running any of the other scripts that operate on the training data.

For further information on the covariates, see the patient questionnaires `Tommy's Net Female Patient Questionnaire v3.0-clean.pdf` and `Tommy's Net Male Patient Questionnaire v3.0-clean.pdf`.

2.2 Test Data

The test data is contained with three files. `rfe003_demobaseline.csv` contains all the covariate information, and `rfe003_pregnancies.xlsx_RCS_NK` contains the pregnancy outcomes. In the former file some patients are missing a date of first consultation, these can be found in `Date of first consultation.csv`.

The script `TestDataSetup.R` contains script to prepare the data for analysis. Again this is probably better done within Excel, but for now this script must be run before running any of the others that operate on the test data. Note that all the testing of the cure rate models is done in R, so there is no corresponding Python script.

3 Basic Analysis

This sections contains details about the scripts used for the Survival Analysis and Logistic Regression analysis.

3.0.1 LogisticRegression.R

Runs a logistic regression using R's inbuilt `glm` function. The response variable is an indicator of whether a woman's first pregnancy following her first consultation is viable (i.e. is successful or is beyond 24 weeks), with 1 representing a viable pregnancy. To run this you need vectors of equal length containing all the covariates and the outcome in 0-1 format.

3.0.2 LRCV.R

This script runs Leave-One-Out Cross Validation (LOOCV) for the Logistic Regression classifier on the training data set. The script returns a confusion matrix and several accuracy metrics: accuracy, sensitivity, specificity, Precision and Negative Predictive Value (NPV).

3.0.3 LRTest.R

Tests the predictions of the Logistic Regression model on the test data set.

3.0.4 LRROC.R

An extension of `LRCV.R` that plots ROC and Precision-Recall curves as the threshold probability varies. By threshold probability, we mean the predicted probability of success above which we predict a successful pregnancy.

3.0.5 ROCCurves.R

Performs the same function as `LRROC.R`, but acts on the test data set.

3.0.6 SurvivalAnalysis.R

Fits a Cox Regression model to the training data using the function `coxph` from the CRAN package `survival`.

3.0.7 PHValidation.R

This script does basic cross validation for the Cox Regression model described above. For details on how this is done, see the full report. Uses the `Surv` function from the `survival` package.

3.0.8 PHTest.R

Tests the predictions of the Cox Regression on the test data set. Uses the `Surv` function from the `survival` package.

4 Cure Rate Models

This section contains information about the scripts used to fit and test the cure rate models.

4.1 Markov Chain Monte Carlo

The scripts in this section are all for running MCMC for different models on the training data. With the exception of `NoCovariatesMCMC.py`, we use a Metropolis-Hastings Algorithm with Normal proposals centered on the current state.

4.1.1 NoCovariatesMCMC.py

This script runs a Gibbs sampler for the model without covariates described in section 5.1 of the summer write up, or section 5.2 of the dissertation:

$$T \sim \begin{cases} \text{Exp}(\beta) & \text{with probability } q \\ \infty & \text{with probability } 1-q \end{cases}$$

It also contains some script to test the predictions of the model.

4.1.2 EventClassMCMC.py

This script runs a MH sampler for the model described in section 5.2 of the summer write up, or section 5.4 of the dissertation:

$$T \sim \begin{cases} \text{Exp}(\beta) & \text{with probability } \frac{1}{1+\exp(-\alpha^T x)} \\ \infty & \text{with probability } 1 - \frac{1}{1+\exp(-\alpha^T x)} \end{cases}$$

There is code at the end that runs multiple chains and stores them in Excel for convergence analysis, make sure to comment this to avoid excessive run times.

4.1.3 EventClassMCMCCV.py

Runs 10-fold cross validation for the model described in `EventClassMCMC.py`. Note that there are slight complications with testing the predictions due to censoring. See the report for further details.

4.1.4 NoCureRateMCMC.py

Runs a MH sampler for the model described in section 5.3 of the summer write up:

$$T \sim \text{Exp}(e^{\beta^T x})$$

There is code at the end that runs multiple chains and store them in Excel for convergence analysis, make sure to comment this to avoid excessive run times.

4.1.5 NoCureRateMCMC.py

Runs 10-fold cross validation for the model described in `NoCureRateMCMC.py`. Note that there are slight complications with testing the predictions due to censoring. See the report for further details.

4.2 Other Scripts

Details on the remainder of the scripts used for analysis of the cure rate models.

4.2.1 MCMCTester.R

This script contains code to evaluate the convergence of the chains produced by the various Python scripts. The Python scripts run several chains and store them in an Excel workbook which this file then reads. Uses the package `coda`. See the report for details. Contains sections for each model so make sure to uncomment the section you want.

4.2.2 CureRateTest.R

Tests the predictions of the Bayesian models on the test data set. Contains sections for each model so make sure to uncomment the section you want.