Predicting readmission of preterm babies
A Generalised Linear Model Approach

Lorenzo Croissant

University of Lancaster
l.croissant@lancaster.ac.uk

November 20, 2017

Abstract

The decline of infant mortality has been an exceedingly successful endeavour of modern medicine. In our data-rich world, modern statistical methods have great potential to be used to further reduce under-5 death rates. In particular this paper scrutinises a simple proof-of-concept method to predict the future health of preterm babies. After a discussion of the medical and mathematical background, an exploration of an available dataset will be provided. Later, a Generalised Linear Model will be employed to make predictions, and an effective algorithm will be designed to select predictors. This algorithm will then be used to get a high quality predictive model. Before closing with a discussion of this method and possible further ones, the model will be refined to check its assumptions and offer improvements.

I. Introduction

The infant mortality rate has been steadily decreasing worldwide since the 1950s according to the World Health organisation. In the United Kingdom for instance, it has fallen from 33 to 3.5 per 1000 live-births[1]. In parallel however, the rate of preterm births has been increasing in almost all the developed world, and has remained at around 7.2%[2]. These trends highlight a requirement of focused research of preterm babies in order to keep working towards a minimal infant mortality rate.

This Public Health policy issue has been the subject of extensive research by scientists, governments, and international organisations. While medical statistics are available and can help guide public policy or understand the likely predictors of negative health outcomes, it appears to the author necessary to also develop predictive models applicable to real life situations. The increase in predictive capability of computational statistics over the last few decades has the potential to revolutionise preventive care.

As an initial evaluation we will begin by examining simple models useful for small datasets, such as regressors. In the future, it would be interesting to use large datasets to implement more robust prediction models like Random Forests, Neural Networks, etc. However, these complex models require large sanitised datasets to be trained effectively. It is not guaranteed that the data infrastructure to support them is available in the health sector, and this motivates our exploration of "small data" methods. We will use an existing dataset that was previously collected.

In this study[1], Langley *et al.* examined the influence of community neonatal services on survival of under-1 premature babies. These services help follow discharged infants with advice, link nurses and home visits to families of preterm babies. The study conducted a robust selection of data sources and achieved a sizeable data-set size with 1488 infants and records of 12 variables. These included the length of stay in a neonatal ICU and whether they were re-admitted in their first-year as responses, along with several covariates. These covariates were collected through a simple question form, and as such appear to be easily collectable from prospective parents for predictive purposes. This reasoning encourages the choice of such a data set for a first examination of prediction methods.

Before we begin our analysis we will review background information in statistics required to understand the models used and present other useful results. Afterwards we will present the results of our exploratory data analysis, and then discuss the fitting of models to the data using our p-DEBASE algorithm. We will ultimately proceed with a model selection process and cleaning of the

[1] According to WHO figures.

data to ensure a most efficient model, and follow this with some final remarks.

II. Background

i. Generalised Linear Models

This paper assumes the reader to have a level equivalent to first year university statistics, and a solid understanding of linear regression. It aims to be accessible to students and professionals in fields requiring statistics, but who have not pursued statistics courses for more than one year. This section aims to bring such a reader up to speed.

In linear regression we fundamentally seek to relate the response variable (Y_i) to a linear combination of covariates ($X := [x_{i,j}]$), i.e.

$$Y_i = X\underline{\beta} + \underline{\epsilon}\,. \tag{1}$$

The generalisation of equation (1) relies on allowing the $g(y_i)$, for some function g, to be related to a linear combination of predictors. This distinction allows us to fit models to data that are not related directly by a linear relation, but where there is still some relationship involving an intercept or a dose-response.

We call g the link function of the model, and we can formulate an analog to equation (1):

$$y_i = g^{-1}(X\underline{\beta}) + \underline{\epsilon}\,. \tag{2}$$

We now need to define that a random variable R belongs to the exponential family if the probability distribution of its outcomes satisfies:

$$f(y|\theta,\phi) = \exp\left[\frac{y\theta - k(\theta)}{\phi} + c(y,\phi)\right] \tag{3}$$

Provided that Y_i are IID random variables that belong to an exponential distribution, and there is $\underline{\beta}$ so that equation (2) is satisfied, we have a valid GLM.

ii. GLM Formulæ

For a random variable Y of the exponential family, we have[2] the mean $E[Y] = \mu = k'(\theta)$ and variance $Var(Y) = \phi k''(\theta)$. Of interest is the fact that unlike the mean, the variance of Y depends on ϕ, which we call the scale parameter, and if ϕ is notably larger than one then there will be over-dispersion. Over-dispersion is not an issue in and of itself, but it requires the use of "quasi"-distributions, as the regular distributions such as Poisson or Binomial will assume ϕ to be 1.

An important measure we need is the deviance of a model $\mathcal{M}$. This is akin to a Sum of Squared Errors in Linear Regression, and the larger it is, the worse the power of our model. For modelled Y_i IID exponential family random variables, with realisations y_i given predictors X we have

$$dev(\mathcal{M}) = \sum_{i=1}^{n} 2\phi\left[l(y_i,\phi) - l(X\hat{\beta},\phi)\right]$$

iii. Logistic Regression

In this study, we are interested in looking at a response variable between 0 and 1. To do this, there is an established model which we will use, called Logistic Regression. Before presenting it, we will look at the distribution that underlies it: the Binoprop distribution. Suppose we wanted to model the proportion of successful outcomes from a Binomial(n,p) random variable. This new random variable $0 \leq Z \leq 1$ would follow a Binoprop(n,p) distribution, which is an exponential distribution[3]. There are three important link functions for the Binoprop case:

$$\begin{cases} \text{logit: } g_1(x) = log\left(\frac{x}{1-x}\right) \\ \text{probit: } g_2(x) = \Phi(x) \\ \text{cloglog: } g_3(x) = log(-log(1-p)) \end{cases}$$

If we model Binoprop data using the logit function as a link function, this regression bears the special name of Logistic Regression. Logistic Regression is a common classifier function for simple classification problems, as it relies on very simple mathematical computations.

[2] Derivations are omitted here.

[3] The derivation is omitted here, but here is the gist of it, should the reader wish to derive it on their own: From the PMF of a Binomial, find the PMF of the Binoprop, then rearrange as an exponential function.

iv. Model Evaluation

Since there is now a non-trivial transformation to be done to relate variables in our model, many usual regression diagnostics are no longer valid, and in particular for logistic regression we will use the following in this paper: Analysis of Deviance and the Hosmer-Lemeshow test.

Analysis of Deviance, or ANODE, is an analogue to ANOVA for nested models in a Linear Regression setting. Suppose we want to compare models $\mathcal{A}$ and $\mathcal{B}$ where $\mathcal{B}$ is nested in $\mathcal{A}$, so that all covariates in $\mathcal{B}$ are in $\mathcal{A}$. Then we can compute the following deviance statistic:

$$\frac{dev(\mathcal{A}) - dev(\mathcal{B})}{\phi} \sim \chi^2_{df_B - df_A}.$$

ANODE is a very useful tool for comparing GLM models. In particular, we can always compare models to the Null model, which assumes that all β are zero and is thus nested in any other model.

A test that is specific to Logistic Regression and is useful for evaluating the quality of its fit, is the Hosmer-Lemeshow test. In this test, we assume that our model is well specified, and if the test indicates a very significant p-value we reject this null hypothesis, meaning our model is not correctly specified. Hosmer-Lemeshow is a parametric test, which segments the quantiles of the predicted values into g[4]. bins and compares the predicted and true outcomes in each bin. Significant differences in the proportions of errors between bins would indicate poor-fit, and would lead to a small p-value. The Hosmer-Lemeshow test is no longer recommended by its creators[3], due to the sensitivity to the value of g, lack of resistance to over-fitting, and lack of power. There are new tests that replace it, such as Hosmer and Lemeshow's Omnibus test, but in this paper we will simply plot the p-value over a range of bin sizes to eliminate the possibility that we reached a conclusion due to sheer chance in the choice of g.

[4]It is recommended to choose $g > p + 1$, where p is the number of parameters in the model.

III. Exploratory Analysis

i. EDA

Since the dataframe we are working on is not comprised of raw data but comes from a previous study, it is of high quality and is highly sanitised. A careful analysis roused no suspicions regarding any data points. Plotting of the data did not lead to the observation of any patterns or other features of interest. Figure 1, for instance, shows that the birthweight data, while skewed, does not have distinct outliers and there is a reasonable density of points all the way up to its maximum.

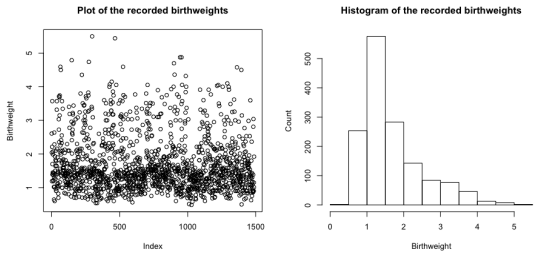

Figure 1: *Exploratory plots for birthweights.*

While the length of stay variable (which is actually the log of the length of stay) seems relatively normal, it is clear from Figure 1 that birthweights are not normally distributed. Birthweights are a purely positive value with a heavy right tail, so we could thus use a log transform to attempt to normalise it. Figure 2 shows that this transform has been successful at normalising the data.

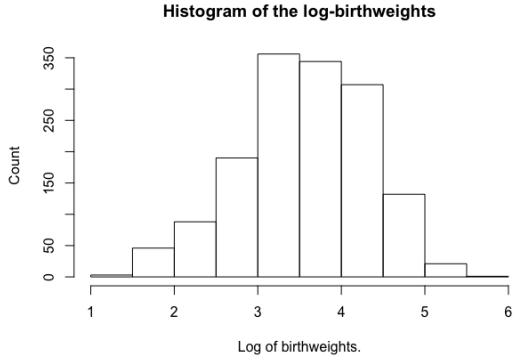

Figure 2: *Result of a log transform applied to birthweights.*

In any classification problem, one must

keep in mind possible class imbalance in the training data. In the case of our data, the classes are well balanced with 44% of babies readmitted. Another issue with any learning problem is dimensionality, which does not appear to be a problem here as our dataset comprises of 1488 datapoints each with only 11 variables[5].

We will now move on to exploration of various modelling possibilities. To do so, our clear objective is to best predict `read`, the factor for readmission, using: continuous variables `bwt` and `los`, the log-transforms of birthweights and lengths of stay, and the factors for employment of the mother, `emp.m`, and of the father, `emp.f`, for the sex of the baby, `sex` (one for male), for the presence of a CNS, `cns`, for the CNS's size, `size`, for the gestation length, `gest` (5 levels), for the mothers age, `age` (4 levels), and for whether or not parents owned their current accommodation.

ii. Multiple Linear Regression

Before we move onto a more complex Generalised Linear Model (GLM) we will outline why regular Linear Regression is not appropriate in the case our problem and data. First of all, our problem is fundamentally classification of a categorical response variable `read`. In Linear Regression, the linear curve that relates the response and predictors, regardless of parameters, will always extend below zero and above one for an uncountably infinite number of points. To ensure that predictions lie in $[0,1]$ for all values of the predictors, which would allow us to build a decision surface, we need to use a non-linear function, such as the sigmoid function[6]. Moreover, Linear Regression would not fit our data either as it does not show a linear relationship between predictors and response, but rather a non-linear relationship.

iii. Empirical GLM Fitting

Since we are interested in predicting the probability of a discharged baby being readmitted, we will use binoprop regression to make our predictions. Before delving deeper into our analysis, we will test out the following three link functions: probit, logit and complementary log-log. In each case we will test the quasi-binoprop model associated, so that we can evaluate dispersion.

Model	Deviance	df	dispersion
`logit`	1914.7	1472	1.01
`probit`	1915.0	1472	1.01
`cloglog`	1910.9	1472	1.01

Table 1: *Summary of link function trials.*

In Table 1, we observe that regardless of the link function chosen we have dispersion that is almost exactly one, which leads us to use regular a binoprop model from now on. We also note that the deviances obtained by the various link functions are almost identical. We will therefore choose the logit link function as it is the canonical link function of the binoprop regression model. From now on we are thus assuming:

$$y_i = \frac{1}{1+\exp\left(-\underline{\beta}\underline{x_i}\right)} + \epsilon_i \,.$$

IV. p-DEBASE for backwards elimination

i. Algorithm

After a presentation of backwards elimination, we will describe the mechanics of p-DEBASE, before observing its results on this problem and then discuss its performance and shortcomings.

When trying to select the best model for a production setting we want to eliminate all the variables that have a negligible impact on prediction. Retaining only the most efficient predictors keeps prediction costs down and can also sometimes lead to improvements in errors. However, it can be a complex and lengthy process to select by hand these valuable covariates. Indeed, they can not reliably be eliminated faster than one at a time. This is due to the way nested model testing works: ANODE determines only whether the simpler model is "better" than the more complex

[5] We excluded the `accom.orig` data about accommodation from our analysis.

[6] This is indeed what we will do in subsection iii. as logit is the inverse of sigmoid.

model it is nested in. If one performs ANODE with a nested model which has five fewer covariates, and the simpler model prevailed, there would be no immediate way to determine which covariates were critical in the test. This could lead to erroneous elimination of a covariate that might have been important.

Backwards elimination, the process of eliminating all unimportant coefficients from the full model, is one method to select the simplest model but not the only one. There is also forward selection, which at each increment of time adds to the model the covariate whose predictive influence is highest, until no covariates are below its selection threshold. This, the reader will see, can lead to issues that resemble the ones above. There are risks of running into local optima in both methods. A slightly more robust method is stepwise selection, which at each step drops the least significant covariate and adds the most significant one not included, until it converges. This method is not perfect either, and model optimisation remains a painful task in building robust production ready models. This motivates the implementation of descent algorithms to more effectively complete the task, much faster than humans could.

p-DEBASE is a simple exemplary set-up for an effective selection algorithm. The effective algorithm would have to be much more complex and much faster than p-DEBASE to be worth implementing in full, and it would probably benefit from being built on stepwise selection rather than backwards elimination. First, we will focus on the heart of the algorithm: "DEBASE", which stands for "Descent for Backwards Selection". This algorithm is, as suggested by the name, an implementation of a descent method for backwards selection of a GLM. Upon which measure descent is practiced would determine the letter that precedes DEBASE, in this case: "p". Our implementation descends along the p-value of individual covariates and eliminates them one by one if they fail to outperform the nested model, which sees them removed, in a simple deviance test.

For implementation purposes, we start with the GLM containing all first order interaction terms g, and a confidence level p. Until we reach our confidence level we eliminate the covariate with the highest p-value, refit the model without it, and repeat. The pseudocode for p-DEBASE is given in Algorithm 1.

Algorithm 1 p-Debase Algorithm

$glm \leftarrow g$
$e \leftarrow 1$
$p \leftarrow 0.05$
while $e \geq p$ **do**
  $glm \leftarrow glm2$
  $e = \texttt{max}\{\texttt{pvalue}(\beta_i)\}$
  $glm2 \leftarrow glm - \texttt{argmax}_{x_i}(e)$
return glm

What we would intuitively desire our algorithm to do, like any efficient descent algorithm, is to, at each step, use the steepest descent direction to pick its next evaluation. It is fairly obvious how a regular Quasi-Newton-Raphson method performs this, but perhaps not so much how this algorithm does. While it has not been explored by the author whether this is the mathematically most optimal descent direction, he would like to provide the intuition that lead to his choice. What we would really desire to do, computational efficiency notwithstanding, would be to at each step, fit all possible models with one less covariate, and then take the one whose p-value in a deviance test is highest. This would identify well which covariate contributed the less to model, but would also force an order of computations that is much higher as the fit of k glm would be required at each step, where k is the current number of covariates remaining in the model. The author sought a measure which is readily available without computations from the fit of the current glm. Once such measure, which is returned by `summary(`glm`)` in `R`, the language used in this instance by the author, is the p-value of each parameter under a simple t-test with $H_0 : \beta_i = 0$. It stands to reason that one can assume that removing the parameter which contributes the least to the model (i.e. that has the highest p-value) has the highest impact on reducing the deviance. This is the assumption that motivates the use of "p-descent" over "D-descent" using deviance.

In section VI. we will discuss further descent methods and other tangent discussions relating to DEBASE.

ii. Results & Discussion

We apply p-DEBASE to our data, with starting model g:

```
g=glm(read~.^2,df,binomial("logit"))
```

Where `df` is the name of the dataframe containing our data. Here g contains all 112 first order interaction, this gives a large number of parameters which can be descended upon, while also presenting patterns that are difficult for humans to analyse. These interaction create a parameter space large enough (but still small compared to $n = 1488$) to allow us to examine the behaviour of p-Debase, and providing a clear motivation to use it rather than perform elimination by hand. Before moving on to the model outputs, let us look at the algorithm's behaviour.

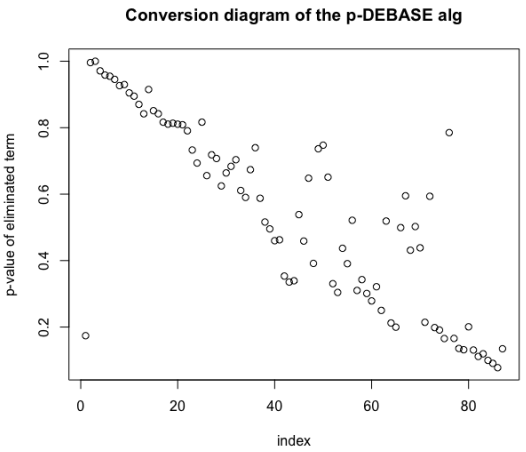

Figure 3: *P-value of the eliminated covariate as the algorithm loops.*

As expected, in Figure 3 there is a strong linear pattern descending from 1 to 0, the interval in which p-values must lie. What is interesting is that the pattern is very distinct towards the starts and ends, but is subject to great variance in the $[20, 75]$ range. It has not been established in this analysis why this is, however there are some elements which could explain it. At the start, the algorithm eliminates the covariates with very high p-values, so who are essentially random with respect to the chance of readmission. At the end, similarly, the model is left with very few independent[7] covariates, and eliminating one has little to no influence on the p-value of the others. However, in the middle, the algorithm might be eliminating highly dependent variables, which might lead to sudden jumps in the maximum p-value, as it is recomputed at each step. With regards to computation time, while it is not optimal, this algorithm still ran in under 30 seconds on a portable computer. Overall, these are satisfactory results.

Covariate	Estimate	p-value
X.Intercept.	-4.3643	0.0000
size1	1.2149	0.0016
bwt	2.2911	0.0147
emp.m1	-0.3101	0.0099
emp.f1	-3.4230	0.0020
age.m2	0.8384	0.0008
los	1.5771	0.0000
cns1.gest4	0.5468	0.0187
cns1.age.m4	-0.6544	0.0417
size1.emp.f1	-0.7963	0.0338
size1.age.m2	-0.6059	0.0131
gest5.bwt	-1.7405	0.0032
gest2.emp.f1	3.5356	0.0019
gest3.emp.f1	3.8744	0.0006
gest4.emp.f1	3.2126	0.0038
gest5.emp.f1	3.5393	0.0005
gest2.los	-0.6148	0.0065
gest3.los	-0.6046	0.0065
gest4.los	-0.5089	0.0290
gest4.sex1	-0.6053	0.0205
gest3.accom1	-0.7706	0.0044
bwt.los	-0.5698	0.0181
emp.m1.age.m4	0.7602	0.0065
age.m2.accom1	-0.4124	0.0379
sex1.accom1	0.5289	0.0002

Table 2: *The raw model output from p-DEBASE*

Table 2 presents the model which has been selected by the p-debase algorithm with $p = 0.05$. As the reader will see, it is a complicated model with 25 covariates and interactions. Most notably, it has selected both continuous

[7] This is not guaranteed, but correlated variables will tend to have lower p-value, as their predictive power will be lowered by their correlated variable.

covariates `bwt` and `los`, their interaction and then many factor interactions. This could be overfitting or it could be due to the identification of complex interaction relationship. See that three levels of `gest`'s five[8] interact with `los`. This is a serious indicator that length of stay and gestational status are interacting. However this line of reasoning fails to well account for terms such as `gest5.bwt`. Is there an interaction only in the last level of gestation, or is this an artefact of the data? Given the large sample size and very small p-value we would suspect that this interaction exists, but a more robust algorithm would need to account for these kinds of over-fitting risks. One simple check to see if our data is fit well by the model would be to compare this model M to the largest nested model containing no interaction terms, $M1$, and to the model $M0$ containing all first order terms and no interaction, found on line "logit" from table **??**, and the null model.

Model	Deviance	df
Null	2062.8	1488
M1	1933.3	1482
M0	1914.7	1472
M	1863.1	1463

Table 3: *Model comparisons using deviance.*

We can see from Table 3, which presents the results of the aforementioned comparison that our model vastly outperforms all three other models. Comparing M1 and M using a deviance test gives us a p-value of the order of 10^{-7}, showing resoundingly that our model has a fit of much higher quality.

We further evaluate this model using the Hosmer-Lemeshow test, and obtain the graph in Figure 4, which shows very clearly that our model fits the data well. Note that for $g = 10$, the standard group size, we have $p \simeq 0.28$.

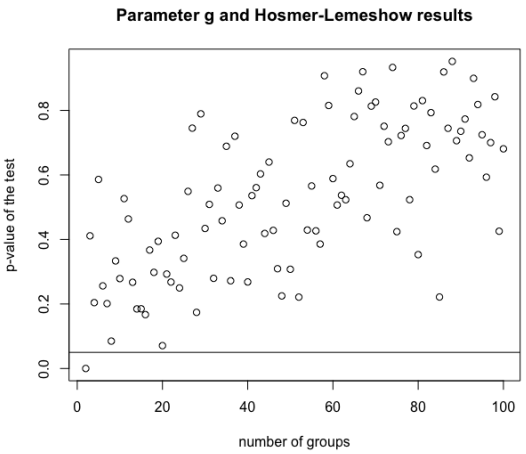

Figure 4: *Sensitivity plot of the Hosmer-Lemeshow test to various group sizes g for our model*

V. Model refining

i. Model Checking

We are limited in logistic regression by the plots that can be used for evaluation of the model. This is unfortunate as it leads mostly to evaluation through statistics, which provide a lot less insight to the reader who is not highly familiar with them. To compensate for this, we will commence the model checking with a slightly unorthodox method for evaluating the dependence of errors on each other and their distribution[9] . In blue on Figure 5, we have a profile of 100 randomly sampled residuals, which has been modified to make it comparable to a gaussian random walk with the same mean and variance, which is drawn in red. There are two reasons for this: first, it allows us to see that there is no correlation between the residual at time t and time $t + 1$, as the blue curve seems to move to a random place from any other place; Second, it allows us to see that the residuals are distributed approximately normally as they have a similar behaviour to a random walk. This by no means a formal test but it will hopefully provide a higher degree of insight than a simple statistic.

After our exhaustive brute force model selection using p-DEBASE, we are now going to use a similarly brute method to evaluate the quality of our model. First off, in order to compute meaningful statistics for our model, we are going to pass the final step and turn it into a true classifier, by adding α, a decision hyper-parameter. The role of alpha is very

[8] Note that the first level `gest1`'s is absorbed by the intercept and not counted as a covariate by our model.

[9] Note that residuals in logistic regression need not be normally distributed.

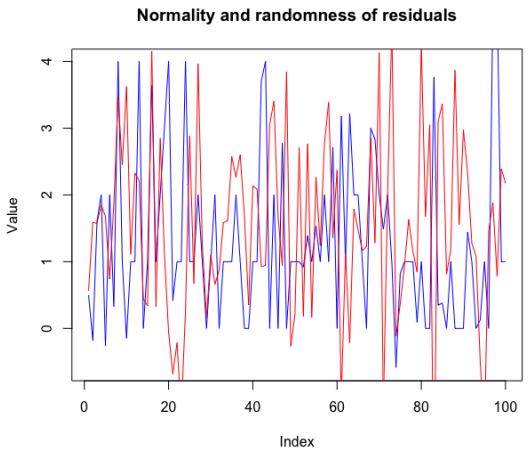

Figure 5: *Normality of residuals*

simple, it allows our model to output only 0 or 1, based on whether it believes the current baby will be readmitted. When presented with the data about a newly discharged baby, instead of outputting a probability, it will now output a class. This is an extension of the discussion we had early on in Section III.iii. about Binoprop vs linear modelling. Only now we output either 0, if the predicted probability p is less than α and 1 if it is above. Since the prior probability of a baby being readmitted, from our EDA in Section III.i, is estimated to be 0.56, we set this as alpha. The Author will not tune the hyper-parameter α, it would not be honest to this analysis if he did, but would rather leave this task to an eventual implementation using a larger data-sample to select the best α based on the medical needs outlined by hospital staff.

Now that we have a true classifier, we can begin to treat it as so and proceed with cross-validation to determine if our model has high sensitivities, and to compute its errors in meaningful ways. We did not use cross validation within p-DEBASE for computational reasons, as cross-validating at each step would be very intensive, even if desirable. Better alternatives to such brute force methods of validation will be discussed further in Section VI.ii. The result of our 100-fold Monte Carlo validation are encouraging and summary statistics can be found below in Table 4

In Table 4, the last two lines present, with

Measure	Mean	Variance
Deviance	1484.8	140.03
Accuracy	62.96%	$6.12e^{-4}$
Predicted 1	25.33%	$6.64e^{-4}$
FP rate	35.88%	$2.87e^{-3}$
FN rate	37.33%	$9.54e^{-4}$
H-L p-val	0.24	0.058
H-L rate	70%	21%

Table 4: *Summary of cross-validated performance.*

$g = 10$, respectively the p-value under the Hosmer-Lemeshow test, and the share of folds where this p-value was greater that 0.05 indicating a good fit at the 5% confidence level. "Predicted 1" denotes the average share of the testing set which was predicted to be readmitted in future.

ii. Outlier Detection and Interpretation

To identify possible outliers we will use Cook's Distance like in Linear Regression. Doing so, we obtain a list of 41 observations with a Distance greater than 4 divided by the degrees of freedom of the model. Eliminating these from our analysis leads us to make a large improvement in the model with a deviance which has fallen from 1863.1 on 1463 degrees of freedom to 1751.8 on 1422 degrees of freedom. For brevity we will not go into further detail about outlier analysis here and we will proceed to interpreting our final model, whose new coefficients are give in table 5

We can interpret any coefficient under logistic regression as the change in log-odds as we change in value of the predictor. Since odds are between 0, and 1 it is easy to see that log-odds lie between $-\infty$ and ∞. So that a negative β_i brings the odds of readmission closer to 0 as x_i increases. For example, an increase of one in the log of the length of stay will increase the log odds of readmission by 2.1252.

Covariate	Estimate	Pr(>|z|)
X.Intercept.	-5.5240	0.0000
size1	1.8009	0.0000
bwt	2.8736	0.0043
emp.m1	-0.3611	0.0035
emp.f1	-4.8371	0.0004
age.m2	1.0796	0.0000
los	2.1252	0.0000
cns1.gest4	0.6056	0.0123
cns1.age.m4	-1.1520	0.0013
size1.emp.f1	-1.2219	0.0039
size1.age.m2	-0.8548	0.0008
gest5.bwt	-2.8197	0.0001
gest2.emp.f1	5.1908	0.0002
gest3.emp.f1	5.5593	0.0001
gest4.emp.f1	4.7118	0.0005
gest5.emp.f1	5.4082	0.0000
gest2.los	-0.9717	0.0005
gest3.los	-0.9595	0.0005
gest4.los	-0.7874	0.0061
gest4.sex1	-0.7647	0.0050
gest3.accom1	-0.7947	0.0043
bwt.los	-0.6914	0.0078
emp.m1.age.m4	0.9406	0.0016
age.m2.accom1	-0.4901	0.0177
sex1.accom1	0.6004	0.0001

Table 5: *The final model of our analysis*

VI. Closing Remarks

i. Limitations of p-DEBASE

We have already spoken about the limitations of p-DEBASE throughout Section IV., but we will summarise and expand upon them here. Firstly, as a descent algorithm it has the typical issues associated with its family of algorithms. While it is not designed for convex continuous optimisation, it still behaves conceptually like a descent algorithm in many ways. Notably, it runs the same risks as gradient descent of getting lodged in a local optimum, without converging to a global optimum. This is fairly simple to resolve in simple cases of convex optimisation as the function can often provide information about the nature of its curvature and allows diagnosis of this issue. However in very high-dimension, or in our case when there is no continuous surface whatsoever it becomes much more difficult to gauge whether or not there are problems of convergence.

The author also would also evaluate computational complexity and speed as a major hinderance to performing this kind of selection effectively. While our implementation has shown relative speed, if we scaled the algorithm to increase its robustness, for example its resistance to outliers, we would necessarily increase the computation complexity and time drastically. As we mentioned earlier, it would be wise to improve our algorithm by setting it to use stepwise selection instead of backwards elimination. Implementing this change could compensate for the increased complexity on problems where the number of features is large but many are not useful as the algorithm would terminate in many fewer steps that the current implementation.

There is an inescapable problem though with a purely numerical descent method. As the model resulting from p-DEBASE showed, the algorithm's idea of the "best" model doesn't necessarily line up with a model which can be interpreted easily by humans. This is a common problem with computational statistics methods, and brings back the eternal bias-variance trade-off, emphasising the role of expert opinions on sensible modelling choices. Sadly, the author did not have such advice at hand whilst writing this paper and as such is limited in the quality of his analysis.

Much like Stochastic Gradient Descent when used in Neural Networks, we suspect that p-DEBASE's use might lead to overfitting. Indeed, in its current form it lacks the cleverness to determine if it is overfitting. We have seen in our model some behaviour which raises suspicions of this. We have seen several factor interactions without the other levels of the factor, which may be the indication of early overfitting. However the author has not pursued robust over-fitting analysis as it is beyond the scope of this paper, preferring instead to focus on cross-validation.

It would be interesting to take a page from the aforementioned Stochastic Gradient Descent and incorporate randomness into our selection process, bagging observations or perhaps even opting for random bagging of covariates on which to apply selection.

These changes would bring us very close to a decision-tree-like structure.

ii. Extension to other Methods

Tree structures are an important part of Data Science and Machine Learning methods. Decision trees, and especially random-forest structures are very effective at decision making on datasets with high complexity and many features. As it was outlined in the introduction, the fundamental problem of predicting the class of an unknown is very heavily studied in Machine Learning, and we restricted ourselves in this paper to only the simplest method, logistic regression. The key difference between Logistic Regression and more complex methods lies in the way their decision surface behaves. In Logistic Regression, we drew a simple linear boundary in the predictor space, using α. Decisions Trees can segment the space of the predictors into many smaller "chunks", each of which will be predicted as either 0 or 1. This simple difference gives tree-based methods and in particular Random Forests an extremely effective way of handling complex interactions, and thus are very effective on large datasets with many predictors and complex underlying mechanisms, like we would expect infant health conditions to have. The other advantage of Random Forests is that they are very fast algorithms with robust behaviour relative to overfitting, thanks to the clever use of random sampling.

Many high-level languages have very effective implementations of Random Forests that are easily deployed into production environments, such as Python's Scikit-Learn library. It stands to reason that these offer prospects for better implementations and more successful evaluation of the risk of readmission of preterm babies, than the elements of this paper's analysis.

References

[1] Langley, D., Hollis, S., Friede, T., MacGregor, D., and Gatrell, A. (2002). Impact of community neonatal services: a multicentre survey. *Archives of Disease in Childhood: Fetal and Neonatal Ed.*,87:F204-F208.

[2] Blencowe, H., Cousens, S., Oestergaard, M., Chou, D., Moller, A., Narwal, R., Adler, A., Vera Garcia, C., Rohde, S., Say, L., Lawn, J.E. (2012) National, regional, and worldwide estimates of preterm birth rates in the year 2010 with time trends since 1990 for selected countries: a systematic analysis and implications *The Lancet*,379:2162-2172

[3] Harrell, F. (2016). Replying to "Evaluating logistic regression and interpretation of Hosmer-Lemeshow Goodness of Fit", *https://www.stats.stackexchange.com*.