Predictive Modelling

Risk Factors Associated with Low Infant Birth Weight

Table of Contents

1) Introduction:

Low birth weight, defined as birth weight less than 2500 grams, is an issue that has been of concern to physicians for years. This is due to the fact that infant mortality and birth defect rates are very high for low birth weight babies. A woman's habits during pregnancy – including diet, smoking, and receiving prenatal care, can greatly alter the chances of delivering a baby of normal birth weight.

Low birth weight babies often are not as strong as a baby of normal birth weight. They may have a harder time eating, gaining weight, and fighting infections. Low birth weight babies also have a hard time staying warm, because they do not have much fat on their bodies (Alliance (UK), 2017).

Data was collected as part of a larger study at Baystate Medical Centre in Springfield, Massachusetts. This dataset contains information on 189 births to women seen in the obstetric clinic; 59 of these births were low weight. The variables identified have been shown to be linked to low birth weight in obstetric studies (W, Lemeshow and Sturdivant, 2013).

In medicine, using predictive modelling helps doctors, staff, and financial departments receive alerts about potential outcomes and risks, so that they are better prepared for the future. This report aims to analyse the risk factors affecting infants' birth weight, and identify which of them are most relevant, by employing predictive modelling techniques.

2) Data Pre-processing

In this section we'll take the necessary pre-processing steps to prepare the dataset for logistic regression, this includes data cleaning, normalisation, splitting, and feature selection.

2.1 Overview

The dataset contains 10 columns, with 4 binary parameters and 189 entries. The variables are:

- low: indicator of birth weight less than 2.5 kg (1 = unhealthy birth weight less than 2.5 kg, 0 = a healthy birth weight).
- age: mother's age in years.
- lwt: mother's weight in pounds at last menstrual period.
- race: mother's race (1 = white, 2 = black, 3 = other).
- smoke: smoking status during pregnancy.
- ptl: number of previous premature labours.
- ht: history of hypertension.
- ui: presence of uterine irritability.
- ftv: number of physician visits during the first trimester.
- bwt: birth weight in grams.

The summary statistics are as follows:

```
> summary(birthwt)
      low              age              lwt        
 Min.   :0.0000   Min.   :14.00   Min.   : 80.0  
 1st Qu.:0.0000   1st Qu.:19.00   1st Qu.:110.0  
 Median :0.0000   Median :23.00   Median :121.0  
 Mean   :0.3122   Mean   :23.24   Mean   :129.8  
 3rd Qu.:1.0000   3rd Qu.:26.00   3rd Qu.:140.0  
 Max.   :1.0000   Max.   :45.00   Max.   :250.0  
      race           smoke            ptl        
 Min.   :1.000   Min.   :0.0000   Min.   :0.0000  
 1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:0.0000  
 Median :1.000   Median :0.0000   Median :0.0000  
 Mean   :1.847   Mean   :0.3915   Mean   :0.1958  
 3rd Qu.:3.000   3rd Qu.:1.0000   3rd Qu.:0.0000  
 Max.   :3.000   Max.   :1.0000   Max.   :3.0000  
       ht                ui        
 Min.   :0.00000   Min.   :0.0000  
 1st Qu.:0.00000   1st Qu.:0.0000  
 Median :0.00000   Median :0.0000  
 Mean   :0.06349   Mean   :0.1481  
 3rd Qu.:0.00000   3rd Qu.:0.0000  
 Max.   :1.00000   Max.   :1.0000  
      ftv              bwt        
 Min.   :0.0000   Min.   : 709  
 1st Qu.:0.0000   1st Qu.:2414  
 Median :0.0000   Median :2977  
 Mean   :0.7937   Mean   :2945  
 3rd Qu.:1.0000   3rd Qu.:3487  
 Max.   :6.0000   Max.   :4990  
> dim(birthwt)
```

```
[1] 189  10
> sum(is.na(birthwt))
[1] 0
```

The dataset does not have any missing values or obvious outliers. Next, we will remove the last column “bwt” which indicates the birth weight in grams – as we won’t need it in this analysis.

```
birthwt1 <- birthwt
birthwt1$bwt <- c()
```

Now we can move onto normalisation with the new subset “birthwt1”.

2.2 Normalisation

Normalisation prior to feature selection is an essential step. As the features are on different scale, one could not compare them with univariate methods until they are one a uniform scale.

2.2.1 Z-score Normalisation:

We will use the Z-score to normalise all the columns except the first, which is the binary variable “low” , which will be the binary outcome for the predictive modelling algorithms.

However, other binary variables such as “smoke” should be normalised – as they are also predictors.

The code and results are as follows:

```
library(robustHD)
for (i in 2:9)
  birthstnd[,i]<-robustHD::standardize(birthstnd[,i])
```

The data is now normalised, we will use a statistical summary function in R to confirm the changes in the mean and standard deviation.

```
> library(psych)
> describe(birthstnd[, c(1:9)])
      vars   n mean   sd median trimmed  mad   min
low      1 189 0.31 0.46   0.00    0.27 0.00  0.00
age      2 189 0.00 1.00  -0.04   -0.06 1.12 -1.74
lwt      3 189 0.00 1.00  -0.29   -0.12 0.68 -1.63
race     4 189 0.00 1.00  -0.92   -0.04 0.00 -0.92
smoke    5 189 0.00 1.00  -0.80   -0.05 0.00 -0.80
ptl      6 189 0.00 1.00  -0.40   -0.24 0.00 -0.40
ht       7 189 0.00 1.00  -0.26   -0.26 0.00 -0.26
ui       8 189 0.00 1.00  -0.42   -0.23 0.00 -0.42
ftv      9 189 0.00 1.00  -0.75   -0.16 0.00 -0.75
       max range skew kurtosis   se
low   1.00  1.00 0.80    -1.36 0.03
age   4.11  5.85 0.71     0.53 0.07
lwt   3.93  5.56 1.38     2.25 0.07
race  1.26  2.18 0.31    -1.75 0.07
smoke 1.24  2.04 0.44    -1.82 0.07
ptl   5.68  6.08 2.76     8.17 0.07
ht    3.83  4.09 3.55    10.67 0.07
ui    2.39  2.81 1.97     1.87 0.07
ftv   4.91  5.66 1.56     3.00 0.07
```

The data is now following a standard normal distribution.

We note that the variable “low” did not change in mean and standard deviation because it was not included in the normalisation.

2.2.2 Null Hypothesis:

In this section we will assess the normality of the distribution.

First, the null-hypothesis will be defines as; the data being normally distributed.

We will use the Shapiro-Wilk test to generate a p-value that can be compared against an existing threshold.

The code and results are as follows:

```
> shapiro.test(birthwt1$low)

      Shapiro-Wilk normality test

data:  birthwt1$low
W = 0.58294, p-value < 2.2e-16

> shapiro.test(birthwt1$age)

      Shapiro-Wilk normality test

data:  birthwt1$age
W = 0.95977, p-value = 3.189e-05

> shapiro.test(birthwt1$lwt)

      Shapiro-Wilk normality test

data:  birthwt1$lwt
W = 0.89331, p-value = 2.242e-10

> shapiro.test(birthwt1$race)

      Shapiro-Wilk normality test

data:  birthwt1$race
W = 0.70831, p-value < 2.2e-16

> shapiro.test(birthwt1$smoke)

      Shapiro-Wilk normality test

data:  birthwt1$smoke
W = 0.61908, p-value < 2.2e-16

> shapiro.test(birthwt1$ptl)

      Shapiro-Wilk normality test
```

```
data:  birthwt1$ptl
W = 0.44704, p-value < 2.2e-16

> shapiro.test(birthwt1$ht)

     Shapiro-Wilk normality test

data:  birthwt1$ht
W = 0.26076, p-value < 2.2e-16

> shapiro.test(birthwt1$ui)

     Shapiro-Wilk normality test

data:  birthwt1$ui
W = 0.42343, p-value < 2.2e-16

> shapiro.test(birthwt1$ftv)

     Shapiro-Wilk normality test

data:  birthwt1$ftv
W = 0.74625, p-value < 2.2e-16
```

As shown by the results of the operation above, all the p-values are extremely small, falling in the range $p \leq 0.001$ – thus, the normality hypothesis is not satisfied for any continuous columns in the dataset. We conclude that the data could not be generated from a normally distributed sample.

Since Shapiro-Wilk test showed that the null hypothesis is not satisfied, we will transform the dataset using a Box-Cox transformation.

```
# Finding lambda.
library(MASS)
transf <- boxcox(birthwt1$lwt ~ birthwt1$age)
lambda <- transf$x[which.max(transf$y)]

# Box-Cox transformation performed on the standardised dataset.
reg1 <- lm(birthwt1$lwt~birthwt1$age)
transf <- boxcox(birthwt1$lwt~birthwt1$age)
lambda <- transf$x[which.max(transf$y)]
reg2 <- lm(((birthwt1$lwt^lambda-1)/lambda)~birthwt1$age)

# Q-Q plots.
par(mfrow = c(1,2))
qqnorm(reg1$residuals)
qqline(reg1$residuals)
qqnorm(reg2$residuals)
qqline(reg2$residuals)
```

Using R functions on two continuous columns, the Box-Cox transformation was performed, and a Q-Q plot was produced. Columns “age” and “lwt” we used.

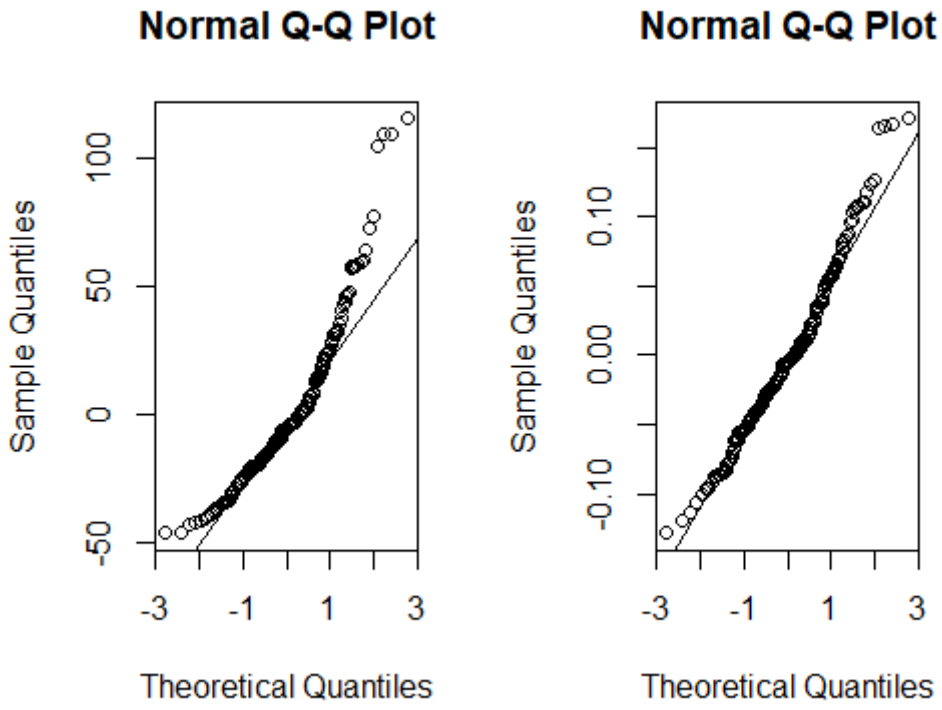

Figure 2.1 – Q-Q normality plot

Figure 2.1 shows that the data now closely follows the line defining the normal distribution. In the left graph, the data points are more scattered, leaning away from the normal distribution. But after applying the Box-Cox transformation, the right graph shows the points closer. In order for data to follow a normal distribution, it should be close to the normal line on the Q-Q plot. So in this case, we can say the transformation worked in getting the data closer to a normal distribution.

Using R, we perform the Shapiro-Wilk test again to check the new p-value.

```
> # Shapiro testing again on transformed values.
> shapiro.test((birthwt1$lwt^lambda-1)/lambda)

      Shapiro-Wilk normality test
data:  (birthwt1$lwt^lambda - 1)/lambda
W = 0.97479, p-value = 0.001715
```

Clearly, the p-value has increased significantly from the previous one, and is now in the range $0.001 < p < 0.01$. This still provides strong evidence against the null hypothesis, so we conclude that the dataset cannot be generated from a normal distribution.

2.3 Splitting

We will proceed with data splitting.

Data splitting is the act of partitioning available data into two portions; one is used to develop a predictive model, and the other to evaluate the model's performance.

It is standard to split data into a “training set” and a “test set”. The reasoning behind this is that it is highly unrealistic to evaluate a system on the data it’s been trained on. The point of a machine learning algorithm is to be able to work with unseen data.

The data will be split into a Training set and a Test set in a 70:30 ratio, with a seed of (134). The code in R is:

```
# Splitting the data.
set.seed(134)
ind <- sample(2, nrow(birthstnd), prob = c(0.7, 0.3), replace =
TRUE)
train.data <- birthstnd[ind == 1,]
test.data <- birthstnd[ind == 2,]
```

And the sets obtained are as follows:

- Training set: 132 entries
- Test set: 57 entries

This proportion is not exactly 70% and 30%. The Training and Test sets are actually about 69.84% and 30.16% respectively (to 2 decimal places).

This can be changed by changing the seed; for example if we use seed of (123) instead, we get:

- Training set: 137 entries (72.49%)
- Test set: 52 entries (27.51%)

Generally speaking, the split hovers close to the wanted accuracy with these seed values. We chose the most accurate seed of (134).

2.4 Feature Selection

Feature Selection is the process of automatically or manually selecting features which contribute most to the prediction variable or output of interest. Having irrelevant features in the dataset can decrease the accuracy of predictive models and allow the model to learn based on irrelevant features, which would not be helpful during application.

Types of feature selection methods are univariate and multivariate.

In this section we will use different methods to select the top 3 relevant features of the birth weight dataset, and form new subsets with these features only.

2.4.1 T-score:

T-score (TSCR) method evaluates a feature using the following criterion

$$J_{ttest}(X_k) = \frac{\mu_1 - \mu_2}{\sqrt{\frac{\sigma_1^2}{n1} + \frac{\sigma_2^2}{n2}}}$$

Where:

- μ_1, and μ_2 are the means of the two classes.
- σ_1^2 and σ_2^2 are the variances of the two classes.
- n_1 and n_2 correspond to the cardinality of the two classes.

The function is first coded in R to be the following:

```
TSCR = function(X, Y, k) # X - matrix with predictors, Y - binary
outcome, k top candidates
{
  J<- rep(NA, ncol(X))
```

```
  names(J)<- colnames(X)
  for (i in 1:ncol(X))
  {
    X1<- X[which(Y==0),i]
    X2<- X[which(Y==1),i]
    mu1<- mean(X1); mu2<- mean(X2)
    var1<- var(X1); var2<- var(X2)
    n1<- length(X1); n2<- length(X2)
    J[i]<- (mu1-mu2)/sqrt(var1/n1+var2/n2)
  }
  J<- sort(J, decreasing=TRUE)[1:k]
  return(list(score=J))
}
```

Then, feature selection is performed.

The X values are set to be the predictors and the Y value is the binary output.

```
# Feature selection on standardised training set.
> library(praznik)
> Data_X<- train.data[,2:9]
> Data_Y<- train.data[,1]
> K<- 3
> TSCR(Data_X, Data_Y, K)
$score
      lwt       age       ftv
2.3386192 0.8256278 0.7783415
```

So according to the T-score, the top 3 most relevant features are the mother's weight at the last menstrual cycle, her age, and the number of physician visits during the first trimester.

2.4.2 Fisher score:

Fisher score (FSCR) method selects features such that the between class distance is maximized and the within class distance is minimized. The scoring criterion is

$$J_{Fisher}(X_k) = \frac{\sum_{i=1}^{2} n_i (\mu_{k,i} - u_k)^2}{\sum_{i=1}^{2} n_i \sigma_{k,i}^2}$$

Where:

- μ_k is the overall mean of the feature.
- X_k , n_i is the number of samples in i^{th} class.
- $\mu_{k,i}$ and $\sigma_{k,i}^2$ is the mean and variance of feature X_k on ith class.

The function is first coded in R:

```
FSCR = function(X, Y, k) # X - matrix with predictors, Y - binary
outcome, k top candidates
{
  J<- rep(NA, ncol(X))
  names(J)<- colnames(X)
  for (i in 1:ncol(X))
```

```
  {
    X1<- X[which(Y==0),i]
    X2<- X[which(Y==1),i]
    mu1<- mean(X1); mu2<- mean(X2); mu<- mean(X[,i])
    var1<- var(X1); var2<- var(X2)
    n1<- length(X1); n2<- length(X2)
    J[i]<- (n1*(mu1-mu)^2+n2*(mu2-mu)^2)/(n1*var1+n2*var2)
  }
  J<- sort(J, decreasing=TRUE)[1:k]
  return(list(score=J))
}
```

Then, feature selection is performed.

The X values are set to be the predictors and the Y value is the binary output in a similar manner to the test above.

```
> FSCR(Data_X, Data_Y, K)
$score
     smoke        ptl        lwt
0.05158048 0.03979784 0.03030525
```

We can see that according to the Fisher score, the top 3 features are whether the mother smoked or not, the number of previous premature labours, and the weight at last menstrual cycle.

2.4.3 Wilcoxon method:

Wilcoxon is a non-parametric method based on ranks for the comparison of the population medians of the two classes defined as

$$J_{Wilcoxon}(X+k) = \frac{\sum_{i=1}^{2} n_i (ur_i - ur)^2}{\sum_{i=1}^{2} \sum_{m=1}^{n_i} (r_{im} - ur)^2}$$

Where:

- N is the total number of samples.
- n_i is the number of samples in class i.
- r_{im} is the rank of sample m in class i.
- μr_i is the average rank of samples belonging to class i.
- μr is the average rank of all samples.

Again, this is translated to code as follow:

```
WLCX = function(X, Y, k) # X - matrix with predictors, Y - binary
outcome, k top candidates
{
  J<- rep(NA, ncol(X))
  names(J)<- colnames(X)
  for (i in 1:ncol(X))
  {
    X_rank<- apply(data.matrix(X[,i]), 2, function(c) rank(c))
    X1_rank<- X_rank[which(Y==0)]
    X2_rank<- X_rank[which(Y==1)]
    mu1<- mean(X1_rank); mu2<- mean(X2_rank); mu<- mean(X_rank)
    n1<- length(X1_rank); n2<- length(X2_rank); N<- length(X_rank)
```

```
    num<- (n1*(mu1-mu)^2+ n2*(mu2-mu)^2)
    denom<- 0
    for (j in 1:n1)
      denom<- denom+(X1_rank[j]-mu)^2
    for (j in 1:n2)
      denom<- denom+(X2_rank[j]-mu)^2
    J[i]<- (N-1)*num/denom
  }
  J<- sort(J, decreasing=TRUE)[1:k]
  return(list(score=J))
}
```

Then, the test is run on the data.

```
> WLCX(Data_X, Data_Y, K)
$score
     ptl    smoke      lwt
8.163830 6.766701 5.017982
```

We can see that according to the Wilcoxon method, the top 3 features are the number of previous premature labours, whether the mother smoked or not, and her weight in pounds at the last menstrual cycle.

From the three tests above, we can see the selected features were “smoke”, “ptl”, “ftv”, “age”, and “lwt”.

For the logistic regression model, we will chose the “lwt”, “age”, and “ftw” variables.

Now, two new subsets were created, the “Reduced Training set” and the “Reduced Test set”, which contain only the top 3 selected variables.

```
reduced.train <- train.data[c(1,9,2:3)]
reduced.test <- test.data[c(1,9,2:3)]
```

The logistic regression will be trained on and fitted to these sets.

3) Data Modelling

3.1 Logistic Regression

Logistic regression is used to model the odds of a certain event in the presence of more than one explanatory variable. The procedure is similar to multiple linear regression, with the exception that the output variable is binomial. The result is the impact of each variable on the odds ratio of the event of interest. The main advantage of this model is to avoid confounding effects, by analysing the association of all variables together (Sperandei, 2014).

Now, we will train the logistic regression on both the Training set and the Reduced Training set. The code in R is as follows:

```
# Logistic Regression for training set.
training_logit <- glm(low~., data=train.data, family="binomial")
summary(training_logit)
# Logistic Regression for reduced training set.
```

```
reduced_logit <- glm(low~., data=reduced.train, family="binomial")
summary(reduced_logit)
```

Now, we will use the Reduced Training set to analyse the 3 predictors we are interested in.

In relation to the birth weight data, we can evaluate the odds of the baby being born with a low body weight depending on certain variables. This is the odds ratio in R and the results are:

```
> # Odds Ratios
> exp(reduced_logit$coefficients)
(Intercept)         ftv         age         lwt
0.4459885   0.8660641   0.8286198   0.6633094
```

Since the intercept is positive, this tells us that the mother being older, having a higher weight, etcetera, increases the odds of the baby being born with a low body weight.

3.2 Random Forest

Random forest is a Supervised Learning algorithm which uses ensemble learning method for classification and regression. It is one of the most used algorithms, because of its simplicity and diversity. It can be used for both classification and regression tasks, It is exceptional in accuracy among current algorithms and runs efficiently on large data sets (Chakure, 2020).

Now, we will train the random forest on both the Training set and the Reduced Training set. The code in R is as follows:

```
# Random Forest for training set.
library(randomForest)
training_forest <- randomForest(factor(low)~., data=train.data,
importance=TRUE, ntree=1000, mtry=3, replace=TRUE)
# Random Forest for reduced training set.
reduced_forest <- randomForest(factor(low)~., data=reduced.train,
importance=TRUE, ntree=1000, mtry=3, replace=TRUE)
```

Now we will look at the importance function; it returns the importance of the features. We will run the code on the Reduced Training set, as it only contains the predictors we are interested in.

```
> importance(reduced_forest)[,c(3,4)]
    MeanDecreaseAccuracy MeanDecreaseGini
ftv            15.060119         7.647152
age             8.523806        20.495073
lwt             6.761479        30.545555
```

The mean accuracy decrease is highest at variable “ftv”, this tells us that the number of physician visits during the first trimester is, according to this data, the most important parameter that indicates whether or not the baby is born underweight. The next most important parameter is the age of the mother, and finally, the weight of the mother at the last menstrual cycle.

3.3 Application of Models

We are now going to apply the Logistic Regression models in section 3.2 to the Test set, and the Reduced Test set.

3.3.1 Applying the Logistic Regression Model:

We will apply the logistic regression model to predict the output using the following code:

```
# Applying the Logistic Regression models to the test set, and the
reduced test set.
prGLM1<- predict(training_logit, newdata=test.data, type="response")
prGLM1
prGLM2<- predict(reduced_logit, newdata=reduced.test,
type="response")
```

To evaluate the performance of the model, let's look at the performance tables.

```
> # Performance tables for the models.
> table(test.data$low, prGLM1>0.5)
    FALSE TRUE
  0    32    5
  1     9    6
> table(reduced.test$low, prGLM2>0.5)
    FALSE TRUE
  0    39    1
  1    16    1
```

The two contingency tables above tell us that the model is performing okay on the Test set, and performing better on the Reduced Test set.

- For the Test set, 32 out of 37 babies were classified correctly as having a healthy body weight. (86.5%)
- For the Reduced Test set, 39 out of 40 babies were classified correctly as having a healthy body weight. (97.5%)

Their performance will be compared further in the coming sections.

3.3.2 Applying the Random Forest Model

We will apply the random forest model to predict the output using the following code:

```
# Applying the Random Forest models to the test set, and the reduced
test set.
prRF1<- predict(training_forest, newdata=test.data, type="response")
prRF2<- predict(reduced_forest, newdata=reduced.test,
type="response")
```

To evaluate the performance of the model, we will look at the performance tables again.

```
> table(test.data$low, prRF1)
    0  1
  0 30  7
  1  7  8
> table(reduced.test$low, prRF2)
    0  1
  0 38  2
  1  2 15
```

The two contingency tables above tell us that the model is performing okay on the Test set, and performing better on the Reduced Test set.

- For the Test set, 30 out of 37 babies were classified correctly as having a healthy body weight. (81.1%)

- For the Reduced Test set, 38 out of 40 babies were classified correctly as having a healthy body weight. (95%)

From these numbers, we deduce that the logistic regression model performed better than the random forest model.

4) Interpretation

4.1 AUC – ROC Curves

In this section, we will create the ROC (receiver operating characteristic) curves for the Test and Reduced Test sets, and evaluate the model's performance by finding the area under the curves.

The ROC curve is a graphical plot of the sensitivity (TP rate) vs 1-specificity (FP rate) for a binary classifier system as its discrimination threshold is varied. The Area Under ROC Curve (AUC) is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one.

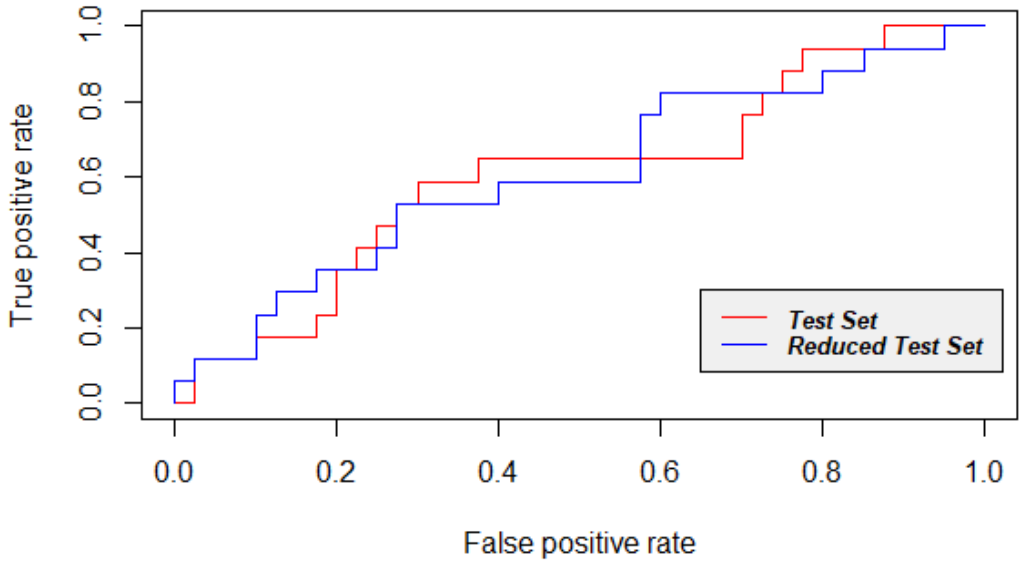

Figure 3.1 - ROC curves

The ROC curves were plotted for the sets after fitting it to the logistic regression model.

The figure above is the ROC-curve for the Reduced Test set. It's composed of 3 elements:

- x-values correspond to [1 – specificity]; they start at 0 and finish at 1.
- y-values correspond to the sensitivity; they start at 0 and finish at 1
- Alpha values correspond to the threshold which covers all values between 0 and 1.

The AUC (area under the curve) of the ROC-curve is a general measure of the usefulness of tests and is used to compare them. The AUC was calculated for this curve using R. The result was as follows:

```
> somers2(prGLM1, test.data$low)
         C        Dxy          n    Missing
 0.7495495  0.4990991 52.0000000  0.0000000
> somers2(prGLM2, reduced.test$low)
```

```
         C         Dxy           n     Missing
 0.6235294   0.2470588 57.0000000   0.0000000
```

We can see from the tables that the area under the curve for the original Test set is higher than the area under the curve for the Reduced Test set. Thus, the feature reduction has decreased the AUC by 16.81%.

4.2 Hosmer–Lemeshow test

We will finally test how well calibrated the logistic regression model is using the Hosmer–Lemeshow test. It is a statistical test for goodness-of-fit for logistic regression models. The test assesses whether or not the observed event rates match expected event rates. Models for which expected and observed event rates are similar are called well calibrated (W, Lemeshow and Sturdivant, 2013).

The formula for the Hosmer–Lemeshow test is:

$$\frac{\sum(observed - expected)^2}{\text{expected}}$$

The code in R is as follows:

```
> hoslem.test(test.data$low, prGLM1)

      Hosmer and Lemeshow goodness of fit (GOF)
      test

data:  test.data$low, prGLM1
X-squared = 8.6585, df = 8, p-value = 0.3719

> hoslem.test(reduced.test$low, prGLM2)

      Hosmer and Lemeshow goodness of fit (GOF)
      test

data:  reduced.test$low, prGLM2
X-squared = 13.297, df = 8, p-value = 0.102
```

From the results, we observe the p-values for both the Test and Reduced Test sets. The p-value are both significant enough to indicate a well calibrated model. Let's look at the goodness-of-fit graphs.

We can see from figure 4.1 that despite the significant p-value, there is lack of fit of the logistic curve to the observed data in the Test set, and an improved fit in the Reduced Test set.

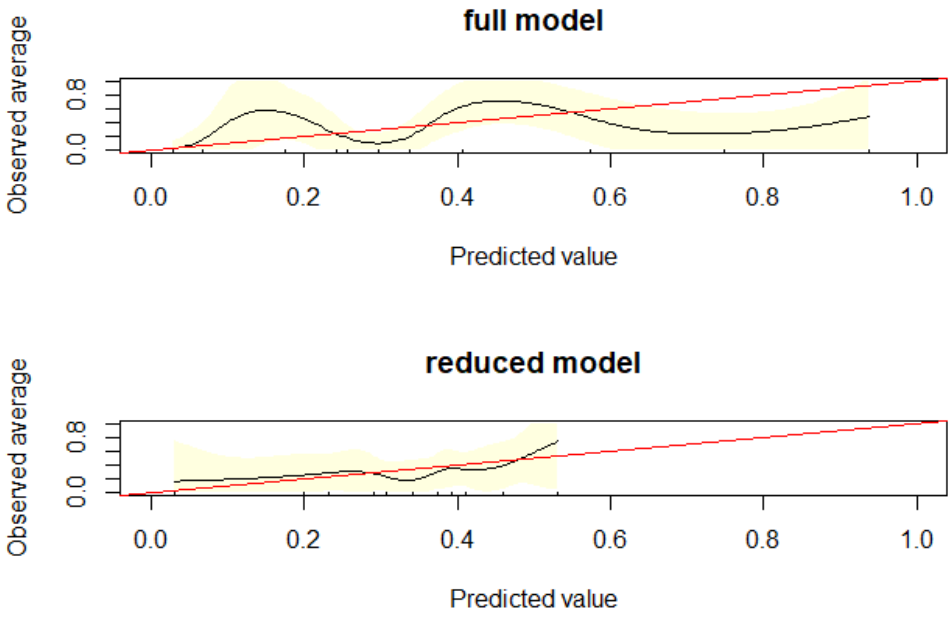

Figure 4.1 – Goodness-of-fit graph

5) Summary of Investigation

This report looked into the dataset "Birthwt" and experimented with logistic regression models in hopes of predicting factors that affect an infant's weight at birth.

The binary variable selected was "low", which indicated the low or normal birth weight of the child. The data was pre-processed and normalised, and the normal distribution was assessed. Then, the dataset was split into a Training set and a Test set, both of which underwent feature selection, from which we went on to produce the Reduced Training and Reduced Test sets. These sets contained the top three most relevant variables to the output; the mother's weight at the last menstrual cycle, her age, and the number of times she visited a physician during the first trimester.

The models created and trained on the Training and Reduced Training sets we the logistic regression and the random forest models. Then, they were applied to the Test and Reduced Test sets. We deduced that for this particular set, the logistic regression model gave better predictions.

The area under the ROC curves decreased by ≈ 17% in the Reduced Test set in comparison to the Test set. Then the Hosmer–Lemeshow test showed that the model is not well calibrated for the Test set, and relatively better calibrated for the Reduced Test set.

6) Conclusion

Low infant birth weight is a concerning problem that affects many, and lack of awareness about the causes and ways to prevent them could be fatal. The good news is; because of the advances in care of sick and premature babies, more babies are surviving despite being born underweight or with other health conditions. However, prevention of preterm births is the best way to stop babies being born underweight.

Prenatal care is a key factor in preventing preterm births and low birthweight in babies. Because maternal nutrition and weight gain are linked with the infant's birthweight, eating a healthy diet, and gaining the proper amount of weight in pregnancy are essential. Mothers should also avoid alcohol, cigarettes, and other drugs, which can contribute to poor fetal growth, and other complications (Alliance (UK), 2017).

By applying predictive modelling and data analysis on such data, the underlying causes can be pinpointed more accurately, and the information generated can be used to give helpful advice to expecting mother, and allows healthcare professionals to devise healthy plans for pregnant women to follow in order to have the safest birth for both them and the child.

* * * * *

Bibliography

Alliance (UK), N.G. (2017). *Risk and prevalence of developmental problems and disorders*. [online] *www.ncbi.nlm.nih.gov*. National Institute for Health and Care Excellence (UK). Available at: https://www.ncbi.nlm.nih.gov/books/NBK533201/ [Accessed 15 Jan. 2021].

Chakure, A. (2020). *Random Forest and Its Implementation*. [online] Medium. Available at: https://medium.com/swlh/random-forest-and-its-implementation-71824ced454f.

Sperandei, S. (2014). Understanding logistic regression analysis. *Biochemia Medica*, 24(1), pp.12–18.

W, D., Lemeshow, S. and Sturdivant, R.X. (2013). *Applied logistic regression*. [online] New York, Etc.: John Wiley And Sons, Cop. Available at: https://www.wiley.com/en-us/Applied+Logistic+Regression%2C+3rd+Edition-p-9780470582473.