Fetal Health Classification in Presence of Unbalanced Data

Sagar Das

M.Sc. Statistics

Semester 4

STSPCOR18M

10815114002220

July, 2022

ACKNOWLEDGEMENT

Date: 22-July-2022

Place: Barasat

First, I wish to express my sincere gratitude to my Professors, Dr. Mrinal Nandi, Dr. Sumanta Adhya, Dr. Sanghamitra Pal, Dr. Sibnarayan Guria for their continuous support and guidance. They provided me with insightful comments and helpful information, practical advice that have helped me at all times throughout this Project. Their immense knowledge, profound experience and professional expertise in the topic of Generalized Linear models, Probability Distribution and Random Sampling have enabled me to understand and proceed with this project successfully. Without their support and guidance, this project would not have been possible. Also, I want to thank Dr. Sumanta Adhya especially for providing me the idea about the great opportunity to work on this topic, and guiding me how to proceed with the problems in this project.

I would also like to thank my parents and friend for their continuous encouragement.

Sagar Das

Semester IV

M.Sc. Statistics

West Bengal State University

Table of Contents

Abstract:

In this project we are dealing with Cardiotocogram data, a device which monitors and records fetal heart rate and uterine contractions during pregnancy and labour. Using this dataset our goal is to correctly classify fetal health of future individuals in order to prevent child and maternal mortality.

We have three classes to classify fetal health of a fetus, Normal, Suspicious and Pathologic, so this is multiclass classification problem. But during exploratory data analysis of this dataset, we found that this dataset is imbalanced in response variable. Furthermore, there exist sparsity in explanatory variables. We would like to solve these problems to classify fetal health of all three classes we have, as correctly as we can. So, we will use various statistical methodologies to try to minimize these issues, and find and appropriate model which will fit this multiclass classification problem. We will use part of our dataset to test our fitted models to see if the fit is good. Then we will draw conclusions according to our findings All the technical part of the project work is done with R.

Introduction:

Reduction of child mortality is one of the most important in several of the United Nations' Sustainable Development Goals and is a key indicator of human progress.
The UN expects that by 2030, countries end preventable deaths of new-borns and children under 5 years of age, with all countries aiming to reduce under-5 mortality to at least as low as 25 per 1,000 live births.

Parallel to notion of child mortality is of course maternal mortality, which accounts for 295 000 deaths during and following pregnancy and childbirth (as of 2017). The vast majority of these deaths (94%) occurred in low-resource settings, and most could have been prevented.

In light of what was mentioned above, Cardiotocograms (CTGs) are a simple and cost accessible option to assess fetal health, allowing healthcare professionals to take action in order to prevent child and maternal mortality. The equipment itself works by sending ultrasound pulses and reading its response, thus shedding light on fetal heart rate (FHR), fetal movements, uterine contractions and more.

This dataset contains 2126 records of features extracted from Cardiotocogram exams, which were then classified by three expert obstetritians into 3 classes:

- Normal
- Suspect
- Pathological

Next, we will take a look at the features of our dataset

We have 22 variables in this dataset, let us briefly go through them.

Our explanetory Variables are:

- **'baseline value'** Fetal Heart Rate baseline (beats per minute)
- **'accelerations'** Number of Fetal Heart Rate accelerations per second
- **'fetal_movement'** Number of fetal movements per second
- **'uterine_contractions'** Number of uterine contractions per second
- **'light_decelerations'** Number of light decelerations of Fetal Heart Rate per second
- **'severe_decelerations'** Number of severe decelerations of Fetal Heart Rate per second
- **'prolongued_decelerations'** Number of prolonged decelerations of Fetal Heart Rate per second
- **'abnormal_short_term_variability'** Percentage of time with abnormal short term variability
- **'mean_value_of_short_term_variability'** Mean value of short term variability
- **'percentage_of_time_with_abnormal_long_term_variability'** Per centage of time with abnormal long term variability
- **'mean_value_of_long_term_variability'** Mean value of long term variability
- **'histogram_width'** Width of Fetal Heart Rate histogram
- **'histogram_min'** Minimum (low frequency) of Fetal Heart Rate histogram
- **'histogram_max'** Maximum (high frequency) of Fetal Heart Rate histogram
- **'histogram_number_of_peaks'** Number of histogram peaks
- **'histogram_number_of_zeroes'** Number of histogram zeros

- **'histogram_mode'** Histogram mode
- **'histogram_mean'** Histogram mean
- **'histogram_median'** Histogram median
- **'histogram_variance'** Histogram variance
- **'histogram_tendency'** Histogram tendency

And our response variable is:

- **'fetal_health'** with three classes as 1 (Normal), 2 (Suspect) and 3 (Pathological)

Our goal is to model this data appropriately with a multiclass classification model to yield class good predictions for future individuals. So, we will use various statistical methodologies to see which models fit our data appropriately, or if there needs to be any statistical techniques to be used to transform the dataset.

Exploratory Data Analysis:

Our data looks like this

baseline value	accelerations	fetal_movement	uterine_contractions	light_decelerations	severe_decelerations	prolongued_decelerations	abnormal_short_term_variability	mean_value_of_short_term_variability	percentage_of_time_with_abnormal_long_term_variability	mean_value_of_long_term_variability	histogram_width	histogram_min	histogram_max	histogram_number_of_peaks	histogram_number_of_zeroes	histogram_mode	histogram_mean	histogram_median	histogram_variance	histogram_tendency	fetal_health
120	0	0	0	0	0	0	73	0.5	43	2.4	64	62	126	2	0	120	137	121	73	1	2
132	0.006	0	0.006	0.003	0	0	17	2.1	0	10.4	130	68	198	6	1	141	136	140	12	0	1
133	0.003	0	0.008	0.003	0	0	16	2.1	0	13.4	130	68	198	5	1	141	135	138	13	0	1
134	0.003	0	0.008	0.003	0	0	16	2.4	0	23	117	53	170	11	0	137	134	137	13	1	1
132	0.007	0	0.008	0	0	0	16	2.4	0	19.9	117	53	170	9	0	137	136	138	11	1	1
134	0.001	0	0.01	0.009	0	0.002	26	5.9	0	0	150	50	200	5	3	76	107	107	170	0	3
134	0.001	0	0.013	0.008	0	0.003	29	6.3	0	0	150	50	200	6	3	71	107	106	215	0	3
122	0	0	0	0	0	0	83	0.5	6	15.6	68	62	130	0	0	122	122	123	3	1	3
122	0	0	0.002	0	0	0	84	0.5	5	13.6	68	62	130	0	0	122	122	123	3	1	3
122	0	0	0.003	0	0	0	86	0.3	6	10.6	68	62	130	1	0	122	122	123	1	1	3
151	0	0	0.001	0.001	0	0	64	1.9	9	27.6	130	56	186	2	0	150	148	151	9	1	2
150	0	0	0.001	0.001	0	0	64	2	8	29.5	130	56	186	5	0	150	148	151	10	1	2
131	0.005	0.072	0.008	0.003	0	0	28	1.4	0	12.9	66	88	154	5	0	135	134	137	7	1	1
131	0.009	0.222	0.006	0.002	0	0	28	1.5	0	5.4	87	71	158	2	0	141	137	141	10	1	1
130	0.006	0.408	0.004	0.005	0	0.001	21	2.3	0	7.9	107	67	174	7	0	143	125	135	76	0	1
130	0.006	0.38	0.004	0.004	0	0.001	19	2.3	0	8.7	107	67	174	3	0	134	127	133	43	0	1
130	0.006	0.441	0.005	0.005	0	0	24	2.1	0	10.9	125	53	178	5	0	143	128	138	70	1	1
131	0.002	0.383	0.003	0.005	0	0.002	18	2.4	0	13.9	107	67	174	5	0	134	125	132	45	0	2
130	0.003	0.451	0.006	0.004	0	0.001	23	1.9	0	8.8	99	59	158	6	0	133	124	129	36	1	1
130	0.005	0.469	0.005	0.004	0	0.001	29	1.7	0	7.8	112	65	177	6	1	133	129	133	27	0	1
129	0	0.34	0.004	0.002	0	0.003	30	2.1	0	8.5	128	54	182	13	0	129	104	120	138	0	3

First, we verified that our data has 22 variables and 2126 observations.

Loading the Dataset

```
data = read.csv('C://Users//sagar//Downloads//fetal_health.csv')
```

```
print(paste("Number of observations:",nrow(data)))
print(paste("Number of variables:",ncol(data)))
```

```
[1] "Number of observations: 2126"
[1] "Number of variables: 22"
```

We find out that three classes of our response variables are distributed

In an unbalanced manner. Among 2126 observations 1655 are Normal, 295 are Suspicious and 176 are Pathological. So, Normal fetuses are 77.85% of all fetuses and Suspicious fetuses are 13.88% and Pathological are 8.28% of all.

```
table(data$fetal_health)
```

```
   1    2    3
1655  295  176
```

```
cbind(c(1,2,3),round((matrix(table(data$fetal_health))/nrow(data))*100,2))
```

A matrix: 3 × 2 of type dbl

1	77.85
2	13.88
3	8.28

We also find out that two of our variables, "**severe_decelerations**" and "**prolongued_decelerations**" has 0 values for most of the observations.

"**severe_decelerations**" has 99.67% has 0 values, and "**prolongued_decelerations**" 91.63% has 0 values. So, these two variables have presence of sparse data.

```
table(data$severe_decelerations)
```

```
   0 0.001
2119     7
```

```
print(paste("Percentage of 0s in severe_decelerations is",round((1-(7/2126))*100,2)))
```

```
[1] "Percentage of 0s in severe_decelerations 99.67"
```

```
corr_mat <- round(cor(data),2)
```

```
table(data$prolongued_decelerations)
```

```
   0 0.001 0.002 0.003 0.004 0.005
1948    70    72    24     9     3
```

```
print(paste("Percentage of 0s in prolongued_decelerations is",round((1948/2126)*100,2)))
```

```
[1] "Percentage of 0s in prolongued_decelerations is 91.63"
```

Next let us find the correlation between our explanatory variables.

We used the following R code to find correlation.

```
corr_mat <- round(cor(data),2)
```

We find out that "Histogram_mean", " Histogram_median", "Histogram_mode", these three explanatory valriables are highly positively correlated.

Also, "Histogram_width" and "Histogram_mean" are highly negatively correlated.

Our output correlation matrix looks like this,

baseline. value	accelerations	fetal_movement	uterine_contractions	light_decelerations	severe_decelerations	prolonged_decelerations	abnormal_short_term_variability	mean_value_of_short_term_variability	ge_of_time_with_abnormal_long_term_vari	mean_value_of_long_term_variability	histogram_width	histogram_min	histogram_max	histogram_number_of_peaks	histogram_number_of_zeroes	histogram_mode	histogram_mean	histogram_median	histogram_variance	histogram_tendency	fetal_health
1	-0.08	-0.03	-0.15	-0.16	-0.05	-0.1	0.31	-0.28	0.29	-0.03	-0.15	0.36	0.28	-0.11	0	0.71	0.72	0.79	-0.13	0.29	0.15
-0.08	1	0.05	0.09	-0.11	-0.04	-0.13	-0.28	0.21	-0.37	-0.14	0.3	-0.15	0.39	0.19	-0.01	0.24	0.27	0.27	0.13	0.03	-0.36
-0.03	0.05	1	-0.07	0.05	-0.01	0.27	-0.1	0.12	-0.07	0.01	0.16	-0.15	0.1	0.16	-0.02	-0.06	-0.09	-0.07	0.18	0	0.09
-0.15	0.09	-0.07	1	0.29	0.01	0.08	-0.23	0.29	-0.31	-0.07	0.14	-0.11	0.12	0.08	0.06	-0.1	-0.19	-0.14	0.24	-0.07	-0.2
-0.16	-0.11	0.05	0.29	1	0.11	0.23	-0.12	0.56	-0.27	-0.24	0.52	-0.55	0.22	0.4	0.24	-0.35	-0.53	-0.39	0.56	0	0.06
-0.05	-0.04	-0.01	0.01	0.11	1	0.01	0.03	0.03	-0.03	-0.04	0.04	-0.07	-0.02	0.01	0.04	-0.22	-0.16	-0.16	0.14	-0.07	0.13
-0.1	-0.13	0.27	0.08	0.23	0.01	1	0.05	0.27	-0.14	-0.23	0.27	-0.28	0.12	0.22	0.06	-0.44	-0.49	-0.44	0.5	-0.22	0.48
0.31	-0.28	-0.1	-0.23	-0.12	0.03	0.05	1	-0.43	0.46	-0.32	-0.26	0.28	-0.11	-0.17	-0.15	0.06	0.07	0.12	-0.15	-0.01	0.47
-0.28	0.21	0.12	0.29	0.56	0.03	0.27	-0.43	1	-0.47	0.07	0.66	-0.62	0.41	0.5	0.27	-0.31	-0.45	-0.34	0.56	-0.07	-0.1
0.29	-0.37	-0.07	-0.31	-0.27	-0.03	-0.14	0.46	-0.47	1	-0.17	-0.45	0.42	-0.28	-0.28	-0.12	0.17	0.22	0.19	-0.28	0.04	0.43
-0.03	-0.14	0.01	-0.07	-0.24	-0.04	-0.23	-0.32	0.07	-0.17	1	0.11	-0.14	0	0.06	0.12	0.07	0.14	0.06	-0.16	0.15	-0.23
-0.15	0.3	0.16	0.14	0.52	0.04	0.27	-0.26	0.66	-0.45	0.11	1	-0.9	0.69	0.75	0.32	-0.16	-0.28	-0.17	0.62	0.12	-0.07
0.36	-0.15	-0.15	-0.11	-0.55	-0.07	-0.28	0.28	-0.62	0.42	-0.14	-0.9	1	-0.3	-0.67	-0.31	0.35	0.49	0.4	-0.55	-0.24	0.06
0.28	0.39	0.1	0.12	0.22	-0.02	0.12	-0.11	0.41	-0.28	0	0.69	-0.3	1	0.52	0.18	0.24	0.19	0.29	0.44	-0.14	-0.05
-0.11	0.19	0.16	0.08	0.4	0.01	0.22	-0.17	0.5	-0.28	0.06	0.75	-0.67	0.52	1	0.29	-0.1	-0.22	-0.12	0.45	0.11	-0.02
0	-0.01	-0.02	0.06	0.24	0.04	0.06	-0.15	0.27	-0.12	0.12	0.32	-0.31	0.18	0.29	1	-0.06	-0.08	-0.05	0.2	0.08	-0.02
0.71	0.24	-0.06	-0.1	-0.35	-0.22	-0.44	0.06	-0.31	0.17	0.07	-0.16	0.35	0.24	-0.1	-0.06	1	0.89	0.93	-0.31	0.42	-0.25
0.72	0.27	-0.09	-0.19	-0.53	-0.16	-0.49	0.07	-0.45	0.22	0.14	-0.28	0.49	0.19	-0.22	-0.08	0.89	1	0.95	-0.4	0.33	-0.23
0.79	0.27	-0.07	-0.14	-0.39	-0.16	-0.44	0.12	-0.34	0.19	0.06	-0.17	0.4	0.29	-0.12	-0.05	0.93	0.95	1	-0.29	0.39	-0.21
-0.13	0.13	0.18	0.24	0.56	0.14	0.5	-0.15	0.56	-0.28	-0.16	0.62	-0.55	0.44	0.45	0.2	-0.31	-0.4	-0.29	1	-0.08	0.21
0.29	0.03	0	-0.07	0	-0.07	-0.22	-0.01	-0.07	0.04	0.15	0.12	-0.24	-0.14	0.11	0.08	0.42	0.33	0.39	-0.08	1	-0.13
0.15	-0.36	0.09	-0.2	0.06	0.13	0.48	0.47	-0.1	0.43	-0.23	-0.07	0.06	-0.05	-0.02	-0.02	-0.25	-0.23	-0.21	0.21	-0.13	1

Problems and Steps taken:

Our goal in this project is to classify fetal health in each of the outcome classes as accurately as we can, so we have to fit a multiclass classification model accordingly. First, we will use random sampling to divide our dataset into two parts, training set and testing set. We have used SRSWOR to draw 60% of the observations and use it as training data, and the remaining 40% of the observations as test data. We will be using the test data to evaluate the models and methodologies we used on our training dataset.

During the exploratory data analysis, we found out that our dataset has problem of Unbalanced dataset, because only 14% and 8% of all observations are of the classes Suspicious and Pathological fetuses. As these two classes have very small number of observations in our dataset, we will get much lesser information about these two classes that the class with most observations, that is the Normal Fetues class with almost 78% of all the observations. That is why we will use an oversampling method to reduce sample disproportion between three outcome classes of our data. The oversampling technique we will be using is called Synthetic Minority Over-sampling Technique (SMOTE).

We have observed that two of our explanatory variables has the presence of sparse data problem, that is most of the observations in that variable has 0 values.

severe_deceleration has 99.67% values as 0 (2119 out of 2126) and

prolonged_deceleration has 91.63% values 0 (1948 out of 2126).

In this kind of data effects of non-zero values get nullified by large number of zeroes. We will try to tackle this problem in two ways, firstly we will to use a dummy variable to take zero values as 0 and non-zero values as 1 and then we will use our multinomial model. Another way is we will fit an L1-Penalized model to reduce the effect the presence of sparse data.

We have also observed that there exist highly correlation between "Histogram_mean", " Histogram_median", "Histogram_mode", 0.95, 0.93 and 0.89 respectively, and Highly negative correlation between "Histogram_width" and "Histogram_mean", which is -0.9

Using highly correlated variables in fitting a generalized linear model may cause multicollinearity. So, we will drop a combination of these features to build few models to multinomial models and check their Akaike Information Criteria (AIC). We will use the model with least AIC to classify test data classes and check misclassification probability for each class.

In the next section we will elaborately describe the statistical techniques we will be using in the Methodologies and Results section.

Theoretical Discussion about needed Statistical Methodologies:

Training and Test Sample generation using Simple Random Sampling Without Replacement:

Simple random sampling (SRS) is a method of selection of a sample comprising of n a number of sampling units out of the population having N number of sampling units such that every sampling unit has an equal chance of being chosen.

SRSWOR is a method of selection of n units out of the N units one by one such that at any stage of selection, any one of the remaining units have the same chance of being selected, i.e. 1/ N.

If n units are selected by SRSWOR, the total number of possible samples are $\binom{N}{n}$

So, the probability of selecting any one of these samples is $\frac{1}{\binom{N}{n}}$

In this project we will use SRSWOR to sample a train dataset from our data, keep the other part of the sample test set to evaluate our methodologies and models fitted. We used simple random sampling because we want each of our observations in our data should have same probability to be in training and test sample. Furthermore, we will be using the Without Replacement because there should not be any overlap between training and testing samples. That is if an observation is in our training sample, then it should not be in the test sample.

Generalized Linear Model:

A generalized linear model (GLM) is a flexible generalization of ordinary linear regression. The GLM generalizes linear regression by allowing the linear model to be related to the response variable via a link function and by allowing the magnitude of the variance of each measurement to be a function of its predicted value.

Generalized linear models have three components: The random component identifies the response variable Y and its probability distribution. The linear predictor specifies the explanatory variables through a prediction equation that has linear form. The link function specifies a function of E(Y) that the GLM relates to the linear predictor.

Random Component: The random component of a GLM identifies the response variable Y and assumes a probability distribution for it. Standard GLMs treat the n observations on Y as independent. We denote those observations by $(y_1, \dots \dots, y_n)$. In many applications, the observations are binary, such as success or failure. More generally, each y_i might be the number of successes out of a certain fixed number of trials. In either case, we assume a binomial distribution for Y. In this project we have three classes as outcome, Normal, Suspicious and Pathological. So, our outcome variable "Fetal_health" follows multinomial distribution — the multicategory generalization of the binomial distribution.

Linear Predictor: The linear predictor of a GLM specifies the explanatory variables. The name reflects that the variables enter linearly as predictors on the right-hand side of the model equation, in the form $\alpha + \beta_1 x_1 + \cdots + \beta_p x_p$.

Statistical inference for the model conditions on the observed values of the explanatory variables, treating them as fixed rather than as random variables.

Link Function: The expected value μ = E(Y) of the probability distribution of Y has a value that varies according to values of the explanatory variables. The third component of a GLM, the link function, specifies a function g that relates μ to the linear predictor as $g(\mu) = \alpha + \beta_1 x_1 + \cdots + \beta_p x_p$. The link function g connects the random component with the linear predictor function of the explanatory variables.

In this case the link function g(μ) will be log $[\frac{p}{1-p}]$ models the log of an odds. It is appropriate when p is between 0 and 1, such as a probability. This is called the logit link function. A GLM that uses the logit link function and response categorical variable has more two outcome classes is called a multinomial logit model.

Multinomial Distribution:

As each individual fetuses in our data have a non-zero probability to be classified in all three of the classes, Normal, Suspicious and Pathological, distribution of the all the observations in our dataset follows multinomial probability distribution.

If a categorical response variable has more than two possible outcomes, when the observations are independent with the same category probabilities for each, the probability distribution of counts in the outcome categories is the multinomial. Let c denote the number of outcome categories. We denote their probabilities by (π_1, π_2,..., π_c), where $\sum_j \pi_j = 1$. For n independent observations, the multinomial prob- ability that y_1 fall in category 1, y_2 fall in category 2, ..., y_c fall in category c, where $\sum_j \pi_j$ = n, equals

$$P(y_1, y_2, \ldots, y_c) = \left(\frac{n!}{y_1! y_2! \ldots y_c!}\right) \pi_1^{y_1} \pi_2^{y_2} \ldots \pi_c^{y_c}$$

Multinomial Logit Model:

The basic model formula pairs each category with a baseline category. This is why multinomial logit models are sometimes called baseline logit models.

Softwares usually sets the last category (c) as the baseline, in which case the baseline-category logits are

$$\log\left(\frac{\pi_j}{\pi_c}\right),\ j = 1, \ldots, c-1$$

For c = 3, for instance, the model uses $\log(\pi_1/\pi_3)$ and $\log(\pi_2/\pi_3)$. Conditional on the response falling in category j or in category c, $\log(\pi_j/\pi_c)$ is the log odds that the response is j.

The baseline-category logit model with an explanatory variable x is,

$$\log\left(\frac{\pi_j}{\pi_c}\right) = \alpha_j + \beta_j x,\ j = 1, \ldots, c-1$$

The model has c − 1 equations, with separate parameters for each. The effects vary according to the category paired with the baseline. These equations determine equations for all pairs of categories. Here, when c = 3, for example,

$$\log\left(\frac{\pi_1}{\pi_2}\right) = \log\left(\frac{\pi_1/\pi_3}{\pi_2/\pi_3}\right) = \log\left(\frac{\pi_1}{\pi_3}\right) - \log\left(\frac{\pi_2}{\pi_3}\right)$$

$$= (\alpha_1 + \beta_1 x) - (\alpha_2 + \beta_2 x)$$

$$= (\alpha_1 - \alpha_2) + (\beta_1 - \beta_2)x$$

This equation has the form $\alpha + \beta x$ with intercept parameter $\alpha = (\alpha_1 - \alpha_2)$ and with slope parameter $\beta = (\beta_1 - \beta_2)$ with p explanatory variables. For each explanatory variable, different logits have different effects. Unless c and p are both small, the model has a large number of parameters. Software for

multicategory logit models fits all (c − 1) equations simultaneously. The baseline category is arbitrary and the same maximum likelihood parameter estimates occur for a pair of categories no matter which baseline category you use.

Oversampling:

Oversampling can be defined as adding more copies of the minority class to obtain a balanced dataset. Oversampling can be a good choice when you don't have a ton of data to work with. It is appropriate when statisticians do not have enough information. One class is abundant, or the majority and the other is rare, or the minority. This technique attempts to increment the size of rare samples to create a balance when the data is insufficient.

There are few oversampling methods like random oversampling, SMOTE etc.

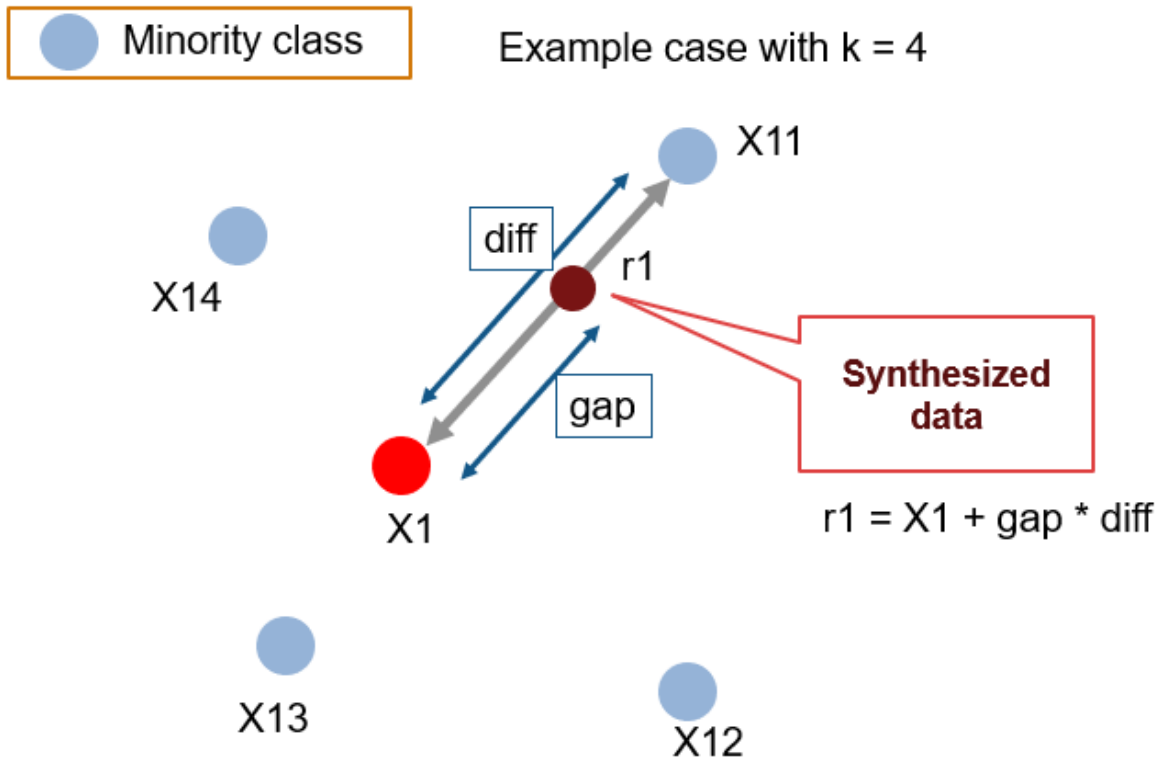
Here in our project to deal with Unbalanced distribution of Response variable "Fetal_Health" we will be using SMOTE oversampling method.

Synthetic Minority Over-sampling Technique:

SMOTE is an oversampling technique where the synthetic samples are generated for the minority class. This algorithm helps to overcome the overfitting problem posed by random oversampling. It focuses on the feature space to generate new instances with the help of interpolation between the positive instances that lie together.

At first the total no. of oversampling observations, N is set up. Generally, it is selected such that the binary class distribution is 1:1. But that could be tuned down based on need. Then the iteration starts by first selecting a positive class instance at random. Next, the KNN's (by default 5) for that instance is obtained. At last, N of these K instances is chosen to interpolate new synthetic

instances. To do that, using any distance metric the difference in distance between the feature vector and its neighbours is calculated. Now, this difference is multiplied by any random value in (0,1] and is added to the previous feature vector. This is pictorially represented below:



For reference, the k-nearest neighbours' algorithm (k-NN) is a non-parametric method first developed by Evelyn Fix and Joseph Hodges. It is used for classification and regression. In both cases, the input consists of the k closest training examples in a data set. The output depends on whether k-NN is used for classification or regression:

In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbours, with the object being assigned to the class most common among its k nearest neighbours (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbour.

Accuracy/ Misclassification measures for Classification Problem:

Here we will describe some of the accuracy/ miscalculation measures we have used in our project.

Classification table:

A classification table cross-classifies the binary outcome y with a prediction of whether y = 0, or 1. The prediction for observation i is $\hat{y}$ = 1 when its estimated probability $\hat{\pi}_i > \pi_0$ and $\hat{y}$ = 0 when $\hat{\pi}_i \leq \pi_0$, for some cutoff π_0. For binary logistic regression one possibility is to take π_0 = 0.50. However, for 3 class cases we can take π_0 = 0.34. So, here if, $\hat{\pi}_j > 0.34$ we can predict $\hat{y}$ = j, j=1,2,3.

Misclassification Probability:

Misclassification probability of j'th class is defined by,

P (the observation is misclassified | observation belongs to j'th class)

Our goal is to minimize this probability when predicting classes for our observations.

Sensitivity and Specificity:

Let us define sensitivity and specificity with and example.

Let X denote the true state of a person (1 = diseased, 2 = not diseased), and let Y = outcome of diagnostic test (1 = positive, 2 = negative).

The accuracy of diagnostic tests is assessed with two conditional probabilities. Given that a subject has the disease (X = 1), the probability the diagnostic test is positive (Y = 1) is called the sensitivity.

Given that the subject does not have the disease (X = 2), the probability the test is negative (Y = 2) is called the specificity.

That is,

$$\text{Sensitivity} = P(Y = 1 \mid X = 1)$$

$$\text{Specificity} = P(Y = 2 \mid X = 2)$$

The higher the sensitivity and specificity, the better the diagnostic test.

Dummy Variable in Linear Models:

We use dummy variable usually when we wish to use categorical variables as predictor variables. Examples include:

Eye colour (e.g., “blue”, “green”, “brown”)

Gender (e.g., “male”, “female”)

Marital status (e.g., “married”, “single”, “divorced”)

When using categorical variables, it doesn’t make sense to just assign values like 1, 2, 3, to values like “blue”, “green”, and “brown” because it doesn’t make sense to say that green is twice as colourful as blue or that brown is three times as colourful as blue.

Instead, the solution is to use dummy variables. These are variables that we create specifically for generalized linear models or ordinary linear models that take on one of two values: zero or one.

The number of dummy variables we must create is equal to k-1 where k is the number of different values that the categorical variable can take on to avoid multicollinearity.

Here to reduce the problem of data sparsity in aforementioned two covariates we first tried taking dummy variables as, when for a given observation in these two variables is 0, dummy variable takes 0 value and when an observation takes non-zero value it takes 1 value.

Let's say for i'th observation sparse variable x_i = 0 then factor I_i = 0

For $x_i \neq 0$, factor I_i = 1

Multicollinearity:

Multicollinearity refers to a situation in which more than two explanatory variables in a multiple regression model are highly linearly related. We have perfect multicollinearity if, for example as in the equation above, the correlation between two independent variables is equal to 1 or −1. In practice, we rarely face perfect multicollinearity in a data set. More commonly, the issue of multicollinearity arises when there is an approximate linear relationship among two or more independent variables.

For linear model $Y_i = \beta_0 + \beta_1 X_{1i} + \cdots + \beta_k X_{ki} + \varepsilon_i.$

The ordinary least squares estimates involve inverting the matrix $X^T X$

Where,

$$X = \begin{bmatrix} 1 & X_{11} & \cdots & X_{k1} \\ \vdots & \vdots & & \vdots \\ 1 & X_{1N} & \cdots & X_{kN} \end{bmatrix}$$

Which is an N × (k+1) matrix, where N is the number of observations and k is the number of explanatory variables (with N required to be greater than or equal to k+1). If there is an exact linear relationship (perfect multicollinearity) among the independent variables, at least one of the columns of X is a linear combination of the others, and so the rank of X (and therefore of XTX) is less than k+1, and the matrix $X^T X$ will not be invertible.

Akaike Information Criteria:

The Akaike information criterion (AIC) is an estimator of prediction error and thereby relative quality of statistical models for a given set of data.[1][2][3] Given a collection of models for the data, AIC estimates the quality of each model, relative to each of the other models. Thus, AIC provides a means for model selection.

AIC is founded on information theory. When a statistical model is used to represent the process that generated the data, the representation will almost never be exact; so, some information will be lost by using the model to represent the process. AIC estimates the relative amount of information lost by a given model: the less information a model loses, the higher the quality of that model.

In estimating the amount of information lost by a model, AIC deals with the trade-off between the goodness of fit of the model and the simplicity of the model. In other words, AIC deals with both the risk of overfitting and the risk of underfitting.

The Akaike information criterion is named after the Japanese statistician Hirotugu Akaike, who formulated it. It now forms the basis of a paradigm for the foundations of statistics and is also widely used for statistical inference.

Suppose that we have a statistical model of some data. Let k be the number of estimated parameters in the model. Let $\hat{L}$ be the maximum value of the likelihood function for the model. Then the AIC value of the model is the following.

$$\mathrm{AIC} = 2k - 2\ln(\hat{L})$$

Given a set of candidate models for the data, the preferred model is the one with the minimum AIC value. Thus, AIC rewards goodness of fit (as assessed by the likelihood function), but it also includes a penalty that is an increasing function of the number of estimated parameters. The penalty discourages overfitting, which is desired because increasing the number of parameters in the model almost always improves the goodness of the fit.

AIC is founded in information theory. Suppose that the data is generated by some unknown process f. We consider two candidate models to represent f: g1 and g2. If we knew f, then we could find the information lost from using g_1 to represent f by calculating the Kullback–Leibler divergence, DKL (f ‖ g_1); similarly, the information lost from using g_2 to represent f could be found by calculating DKL (f ‖ g_2). We would then, generally, choose the candidate model that minimized the information loss.

We cannot choose with certainty, because we do not know f. Akaike (1974) showed, however, that we can estimate, via AIC, how much more (or less) information is lost by g1 than by g2. The estimate, though, is only valid asymptotically; if the number of data points is small, then some correction is often necessary.

Note that AIC tells nothing about the absolute quality of a model, only the quality relative to other models. Thus, if all the candidate models fit poorly, AIC will not give any warning of that. Hence, after selecting a model via AIC, it is usually good practice to validate the absolute quality of the model. Such validation commonly includes checks of the model's residuals (to determine whether the residuals seem like random) and tests of the model's predictions.

Here in this project, we compared AIC of various models to compare them when resolving the multicollinearity issue. And used the model with least AIC among them to classify classes of the outcome variable.

Regularized least squares and Lasso Regression:

In statistics and machine learning, lasso (least absolute shrinkage and selection operator; also Lasso or LASSO) is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model. It was originally introduced in geophysics, and later by Robert Tibshirani, who coined the term.

Lasso was originally formulated for linear regression models. This simple case reveals a substantial amount about the estimator. These include its relationship to ridge regression and best subset selection and the connections between lasso coefficient estimates and so-called soft thresholding. It also reveals that (like standard linear regression) the coefficient estimates do not need to be unique if covariates are collinear.

Though originally defined for linear regression, lasso regularization is easily extended to other statistical models including generalized linear models, generalized estimating equations, proportional hazards models, and M-estimators.

While regularized least squares (RLS) is a family of methods for solving the least-squares problem while using regularization to further constrain the resulting solution.

RLS is used for two main reasons. The first comes up when the number of variables in the linear system exceeds the number of observations. In such settings, the ordinary least-squares problem is ill-posed and is therefore impossible to fit because the associated optimization problem has infinitely many solutions. RLS allows the introduction of further constraints that uniquely determine the solution.

The second reason for using RLS arises when the learned model suffers from poor generalization. RLS can be used in such cases to improve the generalizability of the model by constraining it at training time. This constraint can either force the solution to be "sparse" in some way or to reflect other prior knowledge about the problem such as information about correlations between features. A Bayesian understanding of this can be reached by showing that RLS methods are often equivalent to priors on the solution to the least-squares problem.

The least absolute selection and shrinkage (LASSO) method is another popular choice. In lasso regression, the lasso penalty function R is the ℓ_1 norm, i.e.

$$R(w) = \sum_{j=1}^{d} |w_j|$$

$$\frac{1}{n}\|Y - \mathrm{X}\, w\|_2^2 + \lambda \sum_{j=1}^{d} |w_j| \to \min_{w \in \mathbb{R}^d}$$

In ℓ_1 one can show that for some large enough λ, the solution is a sparse solution. A sparse solution means a solution for w that has many zeros in it, effectively removing the corresponding variable from the system. There are then two immediate consequences:

The system will give a coefficient of zero to non-relevant variables, instead of some insanely small coefficient. This means independent or near-independent variables will be discarded instead of included.

It gives us the most important variables in the regression model. We can speak confidently that variable i has a strong connection with the response variable, Y.

Using Methodologies and results:

At first, we have divided the dataset into training and testing and training SRSWOR. So, these two samples will be non-overlapping and each observation has same probability to be in training or test samples. We will use all the methodologies and models on our training sample and evaluate taken measures with test sample.

Where 60% of our observations will be used training sample and rest of the 40% observations will be used as test sample.

```
train_sample_size = 0.6 * nrow(data)
set.seed(100)
```

```
train_test_sampling = sample(seq_len(nrow(data)), size = train_sample_size, replace = FALSE, prob = NULL)
```

```
train <- data[train_test_sampling, ]
test <- data[-train_test_sampling, ]
```

```
nrow(train)
nrow(test)
```

```
1275
851
```

```
test_data = subset(test, select = -c(fetal_health))
test_result = test["fetal_health"]
```

```
table(test_result)
```

```
fetal_health
  1   2   3
656 126  69
```

We verified that 1275 of our observations as training sample and rest of the 851 observations as test sample.

Also in our test samplen distribution of response class “fetal_health” is 656 are Normal, 126 are Suspicious and 69 are Pathological fetuses.

Model 1:

At first, we will only use multinomial logit on our training dataset and classify for test sample to evaluate the fitted model using misclassification and accuracy measures. We are using nnet R library to perform the multinomial logit model.

```
train$fetal_health <- relevel(as.factor(train$fetal_health), ref = 1)
fit <- multinom(fetal_health ~ ., data = train)
```

```
# weights:  69 (44 variable)
initial  value 1400.730668
iter  10 value 545.109385
iter  20 value 503.040806
iter  30 value 372.186945
iter  40 value 327.576462
iter  50 value 324.650016
iter  60 value 293.790982
iter  70 value 279.462326
iter  80 value 271.387326
iter  90 value 262.462467
iter 100 value 262.461871
final  value 262.461871
stopped after 100 iterations
```

```
predProb = predict(fit, newdata=test_data, type="prob")
```

```
predClass=(colnames(predProb)[max.col(predProb)])
```

```
cm = as.matrix(table(Actual = test_result[,1], Predicted = predClass))
cm
```

The output classification table is

	Predicted		
Actual	1	2	3
1	621	33	2
2	40	76	10
3	8	10	51

```
n = sum(cm)
diag = diag(cm)
```

```
accuracy = sum(diag)/n
accuracy
```

```
0.878965922444183
```

```
missclassification_prob = 1 - accuracy
missclassification_prob
```

```
0.121034077555817
```

```
missclass_1 = 1 - (cm[1,1]/sum(cm[1,]))
print(paste("Missclassification Probablity in Class 1" , round(missclass_1,2)))
missclass_2 = 1 - (cm[2,2]/sum(cm[2,]))
print(paste("Missclassification Probablity in Class 2" , round(missclass_2,2)))
missclass_3 = 1 - (cm[3,3]/sum(cm[3,]))
print(paste("Missclassification Probablity in Class 3" , round(missclass_3,2)))
```

```
[1] "Missclassification Probablity in Class 1 0.05"
[1] "Missclassification Probablity in Class 2 0.4"
[1] "Missclassification Probablity in Class 3 0.26"
```

Here we can see that misclassification probability of our model is 0.12, but if we consider misclassification probability of individual classes, we can see that Normal fetuses has Misclassification Probability of 0.05, Suspicious Fetuses has Misclassification Probability of 0.40 and Pathological Fetuses has Misclassification Probability of 0.26. So, our model can not classify Suspicious and Pathological Fetuses adequately, which is important for fetal and maternal health.

We are dealing with unbalanced distribution of the response variable, so we don't have enough information about Suspicious and Pathological classes. Next up, we are going to introduce SMOTE oversampling method.

Model 2:

In this model we are going to use SMOTE oversampling method to generate sample for Suspicious and Pathological fetuses. In this way we will have enough information about the classes which had very few numbers of observations.

We are using UBL R library to perform the SMOTE oversampling technique.

```
train$fetal_health <- as.character(train$fetal_health)
train$fetal_health[train$fetal_health=="1"]<-"Normal"
train$fetal_health[train$fetal_health=="2"]<-"Suspect"
train$fetal_health[train$fetal_health=="3"]<-"Pathological"
train$fetal_health <- factor(train$fetal_health)
```

```
train <- SmoteClassif(fetal_health ~ ., train, C.perc = list(Normal=1,Suspect=999/169,Pathological=999/107))
```

```
train$fetal_health <- as.character(train$fetal_health)
train$fetal_health[train$fetal_health=="Normal"]<-1
train$fetal_health[train$fetal_health=="Suspect"]<-2
train$fetal_health[train$fetal_health=="Pathological"]<-3
train$fetal_health <- factor(train$fetal_health)
```

```
table(train$fetal_health)
```

```
  1   2   3
999 998 999
```

We can see after applying SMOTE oversampling method we have observations of three classes almost equally distributed.

Now we can apply multinomial logit again to see if the probability of misclassification improved or not.

So, again we will fit multinomial logit model and check classification table and misclassification probabilities of different outcome classes.

We can see that in this model misclassification probability of our model is 0.14, but if we consider misclassification probability of individual classes, we can see that Normal fetuses has Misclassification Probability of 0.12, Suspicious Fetuses has Misclassification Probability of 0.21 and Pathological Fetuses has Misclassification Probability of 0.17.

So, our model improved quite a bit by doing SMOTE oversampling method. As now we can classify Suspicious and Pathological fetuses more correctly.

```
train$fetal_health <- relevel(as.factor(train$fetal_health), ref = 1)
fit <- multinom(fetal_health ~ ., data = train)
```

```
# weights:  69 (44 variable)
initial  value 3291.442417
iter  10 value 2041.291651
iter  20 value 1959.994239
iter  30 value 1420.267299
iter  40 value 1014.347063
iter  50 value 999.008865
iter  60 value 909.045396
iter  70 value 874.859711
iter  80 value 865.225793
iter  90 value 840.271530
iter 100 value 840.259209
final  value 840.259209
stopped after 100 iterations
```

```
predClass = predict(fit, newdata=test_data, type="class")
```

```
cm = as.matrix(table(Actual = test_result[,1], Predicted = predClass))
cm
```

```
      Predicted
Actual   1   2   3
     1 571  73  12
     2  12  99  15
     3   5   7  57
```

```
missclass_1 = 1 - (cm[1,1]/sum(cm[1,]))
print(paste("Missclassification Probablity in Class 1" , missclass_1))
missclass_2 = 1 - (cm[2,2]/sum(cm[2,]))
print(paste("Missclassification Probablity in Class 2" , missclass_2))
missclass_3 = 1 - (cm[3,3]/sum(cm[3,]))
print(paste("Missclassification Probablity in Class 3" , missclass_3))
```

```
[1] "Missclassification Probablity in Class 1 0.129573170731707"
[1] "Missclassification Probablity in Class 2 0.214285714285714"
[1] "Missclassification Probablity in Class 3 0.173913043478261"
```

Model 3:

Here to reduce the problem of data sparsity in aforementioned two covariates, severe_deceleration and prolonged_deceleration, we first tried taking dummy variables as, when for a given observation in these two variables is 0, dummy variable takes 0 value and when an observation takes non-zero value it takes 1 value. Then after taking dummy variables for these two covariates, we are going to fit multinomial logit model again.

```
train_dummy = train
test_data_dummy = test_data
```

```
train_dummy$prolongued_decelerations[train_dummy$prolongued_decelerations==0]<-0
train_dummy$prolongued_decelerations[train_dummy$prolongued_decelerations!=0]<-1
train_dummy$severe_decelerations[train_dummy$severe_decelerations==0]<-0
train_dummy$severe_decelerations[train_dummy$severe_decelerations!=0]<-1
train_dummy$light_decelerations[train_dummy$light_decelerations==0]<-0
train_dummy$light_decelerations[train_dummy$light_decelerations!=0]<-1
```

```
test_data_dummy$prolongued_decelerations[test_data_dummy$prolongued_decelerations==0]<-0
test_data_dummy$prolongued_decelerations[test_data_dummy$prolongued_decelerations!=0]<-1
test_data_dummy$severe_decelerations[test_data_dummy$severe_decelerations==0]<-0
test_data_dummy$severe_decelerations[test_data_dummy$severe_decelerations!=0]<-1
test_data_dummy$light_decelerations[test_data_dummy$light_decelerations==0]<-0
test_data_dummy$light_decelerations[test_data_dummy$light_decelerations!=0]<-1
```

```
train_dummy$fetal_health <- relevel(as.factor(train_dummy$fetal_health), ref = 1)
fit <- multinom(fetal_health ~ ., data = train_dummy)
predClass = predict(fit, newdata=test_data_dummy, type="class")
```

```
# weights:  69 (44 variable)
initial  value 3291.442417
iter  10 value 2041.066955
iter  20 value 1950.023144
iter  30 value 1424.858188
iter  40 value 961.155683
iter  50 value 928.002171
iter  60 value 859.176881
iter  70 value 812.239330
iter  80 value 809.610821
iter  90 value 787.751917
iter 100 value 787.197189
final  value 787.197189
stopped after 100 iterations
```

```
cm = as.matrix(table(Actual = test_result[,1], Predicted = predClass))
cm
```

```
      Predicted
Actual   1   2   3
     1 572  70  14
     2  14  97  15
     3   5  10  54
```

```
missclass_1 = 1 - (cm[1,1]/sum(cm[1,]))
print(paste("Missclassification Probablity in Class 1" , missclass_1))
missclass_2 = 1 - (cm[2,2]/sum(cm[2,]))
print(paste("Missclassification Probablity in Class 2" , missclass_2))
missclass_3 = 1 - (cm[3,3]/sum(cm[3,]))
print(paste("Missclassification Probablity in Class 3" , missclass_3))
```

```
[1] "Missclassification Probablity in Class 1 0.128048780487805"
[1] "Missclassification Probablity in Class 2 0.23015873015873"
[1] "Missclassification Probablity in Class 3 0.217391304347826"
```

We can see that in this model misclassification probability of our model is 0.15, but if we consider misclassification probability of individual classes, we can see that Normal fetuses has Misclassification Probability of 0.12, Suspicious Fetuses has Misclassification Probability of 0.23 and Pathological Fetuses has Misclassification Probability of 0.21.

So, our model did not improve from model 2, where we used multinomial logit model on oversampled training sample, and did not solve sparse data problem.

Model 4:

During exploratory data analysis we observed that some of the covariates of our dataset are highly positively or negatively correlated. So, multicollinearity may occur in our model.

“Histogram_mean”, ” Histogram_median”, ”Histogram_mode”, 0.95, 0.93 and 0.89 respectively, and Highly negative correlation between “Histogram_width” and “Histogram_mean”, which is -0.9

That is why we will build few models where subset of these models is absent, and compare their AIC value to see how much information we are sacrificing.

We will take the model with the least AIC value and classify using that model and evaluate that model using our test sample.

We will build 12 models:

i) histogram_mean-histogram_median-histogram_width is absent
ii) histogram_mode-histogram_median-histogram_width is absent
iii) histogram_mean-histogram_mode-histogram_width is absent
iv) histogram_mean-histogram_median-histogram_min is absent
v) histogram_mode-histogram_median-histogram_min is absent
vi) histogram_mean-histogram_mode-histogram_min is absent
vii) histogram_mean-histogram_median is absent
viii) histogram_mean-histogram_mode is absent
ix) histogram_mode-histogram_median is absent
x) histogram_mean is absent
xi) histogram_median is absent
xii) histogram_mode is absent

So, we fit the following models,

```
model1 <-multinom(train$fetal_health ~.-histogram_mean-histogram_median-histogram_width,data=train)
model2 <-multinom(train$fetal_health ~.-histogram_mode-histogram_median-histogram_width,data=train)
model3 <-multinom(train$fetal_health ~.-histogram_mean-histogram_mode-histogram_width,data=train)
model4 <-multinom(train$fetal_health ~.-histogram_mean-histogram_median-histogram_min,data=train)
model5 <-multinom(train$fetal_health ~.-histogram_mode-histogram_median-histogram_min,data=train)
model6 <-multinom(train$fetal_health ~.-histogram_mean-histogram_mode-histogram_min,data=train)
model7 <-multinom(train$fetal_health ~.-histogram_mean-histogram_median,data=train)
model8 <-multinom(train$fetal_health ~.-histogram_mean-histogram_mode,data=train)
model9 <-multinom(train$fetal_health ~.-histogram_mode-histogram_median,data=train)
model10 <-multinom(train$fetal_health ~.-histogram_mean,data=train)
model11 <-multinom(train$fetal_health ~.-histogram_median,data=train)
model12 <-multinom(train$fetal_health ~.-histogram_mode,data=train)
```

The AIC of the corresponding models are

```
[1] 1952.14
[1] 1939.528
[1] 1807.73
[1] 1952.68
[1] 1933.126
[1] 1807.955
[1] 1948.63
[1] 1810.491
[1] 1939.514
[1] 1800.001
[1] 1824.373
[1] 1770.459
```

So, we are using the xii) and the iii) model to use (Best AIC model and best AIC/ overfitting trade off model).

```
predClass <- predict(model12, newdata=test_data, type="class")
```

```
cm = as.matrix(table(Actual = test_result[,1], Predicted = predClass))
cm
n = sum(cm)
diag = diag(cm)
accuracy = sum(diag)/n
accuracy
```

```
      Predicted
Actual   1   2   3
     1 570  76  10
     2  12  99  15
     3   5   8  56
```

0.851938895417156

```
missclass_1 = 1 - (cm[1,1]/sum(cm[1,]))
print(paste("Missclassification Probablity in Class 1" , missclass_1))
missclass_2 = 1 - (cm[2,2]/sum(cm[2,]))
print(paste("Missclassification Probablity in Class 2" , missclass_2))
missclass_3 = 1 - (cm[3,3]/sum(cm[3,]))
print(paste("Missclassification Probablity in Class 3" , missclass_3))
```

```
[1] "Missclassification Probablity in Class 1 0.13109756097561"
[1] "Missclassification Probablity in Class 2 0.214285714285714"
[1] "Missclassification Probablity in Class 3 0.188405797101449"
```

```
predClass <- predict(model3, newdata=test_data, type="class")
cm = as.matrix(table(Actual = test_result[,1], Predicted = predClass))
cm
n = sum(cm)
diag = diag(cm)
accuracy = sum(diag)/n
accuracy
missclass_1 = 1 - (cm[1,1]/sum(cm[1,]))
print(paste("Missclassification Probablity in Class 1" , missclass_1))
missclass_2 = 1 - (cm[2,2]/sum(cm[2,]))
print(paste("Missclassification Probablity in Class 2" , missclass_2))
missclass_3 = 1 - (cm[3,3]/sum(cm[3,]))
print(paste("Missclassification Probablity in Class 3" , missclass_3))
```

```
      Predicted
Actual   1   2   3
     1 566  80  10
     2  13  96  17
     3   6   9  54
```

```
0.841363102232667
```

```
[1] "Missclassification Probablity in Class 1 0.13719512195122"
[1] "Missclassification Probablity in Class 2 0.238095238095238"
[1] "Missclassification Probablity in Class 3 0.217391304347826"
```

We can see that both of these models perform worse that our model 2, where we performed oversample and fitted multinomial logit model.

Model 5:

Finally, we have used L1 penalized regression using glmnet R library.

```
library(glmnet)
fit <- cv.glmnet(as.matrix(train[,-22]), as.matrix(train[,22]),
                 alpha = 1,family = "multinomial",type.multinomial= "grouped",
                 type.measure = "mse",nfold = 10)
```

```
fit$lambda.min
```

0.000411265198671516

```
y_multi_pred_class <- as.numeric(predict(fit, newx = as.matrix(test_data), type = "class", s = fit$lambda.min))
```

```
coef = coef(fit, s = fit$lambda.min)
```

```
coef
```

Where our coefficients are as follows,

	1
(Intercept)	1.814642e+01
baseline.value	-1.033401e-01
accelerations	1.035082e+03
fetal_movement	-1.277061e+01
uterine_contractions	2.096028e+02
light_decelerations	-4.322816e+01
severe_decelerations	2.603577e+03
prolongued_decelerations	-2.263832e+03
abnormal_short_term_variability	-9.870171e-02
mean_value_of_short_term_variability	9.214343e-01
percentage_of_time_with_abnormal_long_term_variability	-3.093267e-02
mean_value_of_long_term_variability	6.174383e-02
histogram_width	.
histogram_min	-1.069271e-02
histogram_max	-5.715066e-02
histogram_number_of_peaks	1.035524e-01
histogram_number_of_zeroes	-5.985783e-02
histogram_mode	9.866698e-02
histogram_mean	-9.193065e-02
histogram_median	7.670311e-02
histogram_variance	-3.776950e-02
histogram_tendency	-6.185731e-01

We can observe that we have 0 coefficient for Histogram_width, so that variable is not important in classification problem. Also, we have really high coefficient on our sparse data variables severe_decelerations and prolonged_decelerations, giving importance to our non-zero values.

Classification table and misclassification probabilities of this model are as follows.

```
cm = as.matrix(table(Actual = test_result[,1], Predicted = y_multi_pred_class))
cm
```

```
      Predicted
Actual   1   2   3
     1 574  75   7
     2  12 100  14
     3   3  10  56
```

```
n = sum(cm)
diag = diag(cm)
accuracy = sum(diag)/n
accuracy
```

```
0.857814336075206
```

```
missclass_1 = 1 - (cm[1,1]/sum(cm[1,]))
print(paste("Missclassification Probablity in Class 1" , missclass_1))
missclass_2 = 1 - (cm[2,2]/sum(cm[2,]))
print(paste("Missclassification Probablity in Class 2" , missclass_2))
missclass_3 = 1 - (cm[3,3]/sum(cm[3,]))
print(paste("Missclassification Probablity in Class 3" , missclass_3))
```

```
[1] "Missclassification Probablity in Class 1 0.125"
[1] "Missclassification Probablity in Class 2 0.206349206349206"
[1] "Missclassification Probablity in Class 3 0.188405797101449"
```

Comparing this model's misclassification probability with model 2, we can say that this model can classify Normal and Suspicious fetuses better, and Pathologic fetuses almost similarly.

Conclusion:

After using all these models and methodologies we can conclude that, even though model 1, where we just fitted model yielded better overall accuracy,

It cannot classify Suspicious (40% misclassification) and Pathological (28% misclassification) Fetuses adequately, which is our most import goal in this project because fetal and maternal health depends on it. We can also conclude that the higher overall accuracy is a result of presence of Unbalanced distribution in the Response variable fetal_health.

In model 2, where we applied multinomial logit model after performing SMOTE

Oversampling, we can see that misclassification rate of Suspicious and Pathological fetuses reduced quite a bit (21% and 17% respectively).

So, when we have adequate information about these classes, we can detect them better, so SMOTE oversampling helped.

In model 3, to overcome sparsity of two variable sever_deceleration and prolonged_deceleration we used dummy variable. When for a given observation in these two variables is 0, dummy variable takes 0 value and when an observation takes non-zero value it takes 1 value.

But this increased misclassification probability higher with respective to Model 2 for Suspicious and Pathological fetuses. (23% and 21% respectively)

So, taking dummy variable did not resolve the problem of sparsity in covariates.

In model 4 due to correlation of aforementioned variables, we have taken multiple models with presence-absence combination of these variables and compared their AIC. We took the best two models and using the test data we saw these two models are not improvement either as misclassification probability of neither of the classes improved.

In Model 5, our L1 penalized model performed better than our model 2 (SMOTE- Multinomial model), as we can see misclassification probability of Normal and Pathologic fetuses improved here, also misclassification probability of Suspicious fetuses remained almost same.

Here misclassification probabilities of our 3 classes Normal, Suspicious and Pathological fetuses are 0.12, 0.20 and 0.18 respectively.

With high coefficient for variable with sparsity present, this model reduced the effect of sparsity and low coefficient for some variables, negated the unnecessary variables form the model.

So, we can conclude that this model is best fitted amongst the other we displayed in this project.

References

- An Introduction to Categorical Data Analysis by Alan Agresti
- Linear models with R by Julian J Faraway
- Ayres de Campos et al. (2000) SisPorto 2.0 A Program for Automated Analysis of Cardiotocograms. J Matern Fetal Med 5:311-318 for Fetal Health data

- https://www.r-project.org/other-docs.html for help related to R
- https://www.wikipedia.org/ for various research purposes