MALIS Project Report

Prediction of Fetal Risk using Cardiotocogram Data

Amassaghrou Abdelali, Senane Zineb, Zheng Estelle

Motivation

Reduction of child mortality is reflected in the Sustainable Development Goals which aim to reduce, by 2030, neonatal mortality to at least as low as 12 deaths per 1,000 live births and under-5 mortality to at least as low as 25 deaths per 1,000 live births. This objective was already in the Millennium Development Goals but it was not able to meet its target. There is still high child mortality all over the world while most child deaths could have been prevented.

Cardiotocography (CTG) is a technique used to monitor uterine contractions and fetal heartbeat and has been recognized as a prominent indicator of fetal health. Thus, CTG data is beneficial for obstetricians to identify fetal abnormalities and to better decide for medical intervention before persistent damage to the baby. But the analysis of medical data by a doctor could not be objective and correct. Therefore, more and more decision support systems are used to diagnose or predict abnormal situations in many medical fields. We will use machine learning models to identify fetal abnormalities by using the CTG dataset in order to anticipate fetal risks.

We work on CTG data of 2126 pregnant women that were obtained from the University of California Irvine Machine Learning Repository [1]. The dataset is also available on Kaggle [2]. This dataset contains 2126 records of 21 features extracted from Cardiotocogram exams, which were then classified by three expert obstetricians into 3 classes. There is already some research on this topic, one of them is “Use of Machine Learning Algorithms for Prediction of Fetal Risk using Cardiotocographic Data” [3] published in the IJABMR. The results of this study show that Random Forest can be accepted as a good classifier of normal and pathological classes of the CTG data.

Methods

To perform the fetal health prediction, we applied and compared classical machine learning techniques for classification: Logistic Regression, KNN, SVM, Decision tree, Random forest, MLP, Gradient Boosting, and Extreme Gradient Boosting. Meanwhile, we did a features selection analysis and applied these models after performing features selection to verify features' importance.

We will evaluate our models with the recall, the precision, the F1 score, and the accuracy of the classifiers and display them into confusion matrices. Among those metrics, we prioritized the metric recall as there is a high cost associated with false negatives. It ensures that the model correctly detects distressed fetuses who do have the condition.

Data Analysis and features selection

As a first step, we looked at our dataset and preprocessed our data by removing null and duplicate rows. We analyzed the dataset by looking at the distributions of our 3 labels (Normal, Suspect, Pathologic). As the distribution was unbalanced, we decided to combine two labels (Suspect and Pathologic) in one class named Distressed and deal with a binary classification problem. Hence we trained and tested our models by using only the new classes (Normal and Distressed corresponding respectively to 0 and 1). In order to apply our models, we splitted samples into a 70/30 train/test split, and scaled our training set by standardization in order to assure a flawless performance of the classifiers models. We performed cross-validation while training our models.

To perform a selection of features that have a strong impact on the health of the fetus, we applied different methods to understand which features are more important.
Firstly, we started by looking at the correlations of all the features with our target (Normal or Distressed i.e. 0 or 1) using a correlation matrix displayed with a heatmap. Heatmap makes it easy to identify which features are most related to the target variable and to check for collinearity between variables. All the values are between -1 and 1, and the more the absolute value is close to 1, the more the feature is correlated to the target. Thus, features with the lowest values can be dropped and we were able to select 10 best features out of 21.
Secondly, we performed feature selection with a filter method, the univariate selection with ANOVA (analysis of variance) as ANOVA is one of the well-known methods to select the most important features for a classification problem where the inputs have numerical values. We selected the 10 features with the higher scores and we noticed that those 10 features were exactly the same as the ones obtained by the heatmap analysis.
Thirdly, we also tried feature selection using a wrapper method: recursive feature elimination with cross-validation (RFECV). In this method, features are selected by recursively taking a smaller and smaller subset of features by comparing the performance of a ML algorithm as evaluation criteria to select features. We tried RFECV with Logistic Regression and Random Forest. With Logistic Regression, we were able to select 7 features among which 4 were also in the selected features with ANOVA, whereas with the Random Forest, we extracted 11 most important features that included the 10 from ANOVA features selection analysis.
Finally, we applied the decision tree algorithm to select features as it is an intrinsic method of features selection. Among the 10 most important features from this method, 8 were in the 10 best features obtained with ANOVA.
As some features were obtained by each method of feature selection, it helps us understand what features have the most importance with fetal health. However, because the dataset used is a public dataset whose features have already been selected and because we only have 21 features, we worked with both selected features or without.

Experiments and results
Firstly, we applied the models to the whole dataset without performing any features selection. Secondly, we applied all the methods on a new dataset created after performing a features selection by using a heatmap. The preprocessing step is done once we have split the dataset and scaled inputs. All classifiers were implemented using sci-kit-learn.

1. Logistic Regression
For this model, we tested its performance on two different cases. First, we applied it to the scaled data without any polynomial relation between the features. We had an accuracy of 0.89 for testing. Second, we applied a PolynomialFeatures with different orders (from 1 to 7) to the scaled data before even training the model. For orders less than 4, as the order increases by 1, the accuracy increases by 1% until reaching an accuracy of 92% and a recall of 81%. Once we used the 5th order for the polynomial relationship between the features, we reached a higher accuracy and recall (95% and 85% respectively) for testing. When we used an order larger than 5 we saw that the accuracy went down again. These results show us the influence of the order used for the Polynomial features on the performance of the model Logistic Regression. We can explain this influence by the overfitting and underfitting problems caused by the polynomial logistic regression. By looking also at the confusion matrix which is for our case the most important metrics to measure the performance, we find that for Polynomial Logistic Regression with order 5 has the best performance, it has the highest value in predicting correct class and also the lowest value of predicting a normal fetus while it's not normal 0.019.

After this first analysis, we used a K Fold cross-validation with k=5, to test the performance of our model. With cross-validation, the best model to choose is Logistic Regression with a polynomial relation of order 2 between the features. This model has the highest accuracy (93.2%) and recall (82.2%) compared to the others. As the polynomial relation between all the features (21) with an order greater than 2 generates new features, which increase the complexity of the computations. While trying to run the K-Fold cross-validation algorithm for order 5 we found a problem. We were unable to perform these computations as the number of iterations is limited.

2. KNN

For the K-Nearest neighbor model, we applied it first to two cases without cross-validation. The first one is for k=5, we had an accuracy of 92% for both training and testing. The second one is for k=11, we had the same results as the last one. So, we constructed two graphs where we tried to plot the evolution of both the accuracy score and recall score. The results showed that a KNN model with a k=7 is the most suitable for this problem since it gives the best combination of the highest scores of the accuracy and the recall, where the results were 91.8% for accuracy and 75.7% for recall.

Secondly, when applied, the cross-validation has improved the accuracy for the model with k=7 but it has lowered the recall score. Facing this problem, we decided again to plot three graphs that plotted first of all the accuracy score, then the mean square loss score, and finally the recall score.

In this case, the second graph was just a verification process for the first graph.

The best combination of the graph's results is to choose the k=5. Thus the results were that we had an accuracy of 91.6% and a recall of 74%.

But in general, the improvements that were due to the cross-validation weren't important for the KNN model,, the only necessity that it brings is to change the k chosen to get better results.

3. SVM

For the SVM classifier, we tested on two types of the kernel: linear and RBF (radial basis function). First, we applied the models to the scaled data without features selection. For both models, we performed grid search cross-validation to find the best hyperparameters (gamma and C). We considered recall as the scoring for the grid search as it is the most important evaluation metric in our case. For the SVM with a linear kernel, the model performed quite well. It reached an accuracy of 92% during training and 89% during testing. The recall was still good in the testing with a value of 76%. Thanks to cross-validation, the best hyperparameters found were C=10, gamma=0.01. As for the SVM with an RBF kernel, the model performed even better. It reached an accuracy of 98% during training and 93% during testing. The recall also improved: it had a value of 82% during testing. Thanks to cross-validation, the best hyperparameters found were C=10, gamma=0.05.

4. Decision tree

For the decision tree classifier, we started by finding the best hyperparameters using the grid search cross-validation, and then we trained the model to do the prediction. The model performed well as it reached an accuracy of 99% during training and 92% during testing. The recall was also good: it had a value of 83% during testing.

5. Random forest

For the random forest classifier, we also started by finding the best hyperparameters using the grid search cross-validation, and then we trained the model to do the prediction. The model performed really well during training and well during testing. It reached an accuracy of 99.9% during training and 95% during testing. The recall was also high: it had a value of 99.7% during training and 86% during testing.

6. MLP

For the MLP Classifier, we tested two cases: without and with hyperparameters tuning. First, we applied the model to our scaled data and we got an accuracy of 92.4% and a recall of 82.1%, whereas after using the tuned parameters given by the Grid Search we got a lower recall (92.4% for the accuracy and 80.2% for the recall). Hence the grid Search didn't ameliorate the MLPClassifier for this prediction problem.

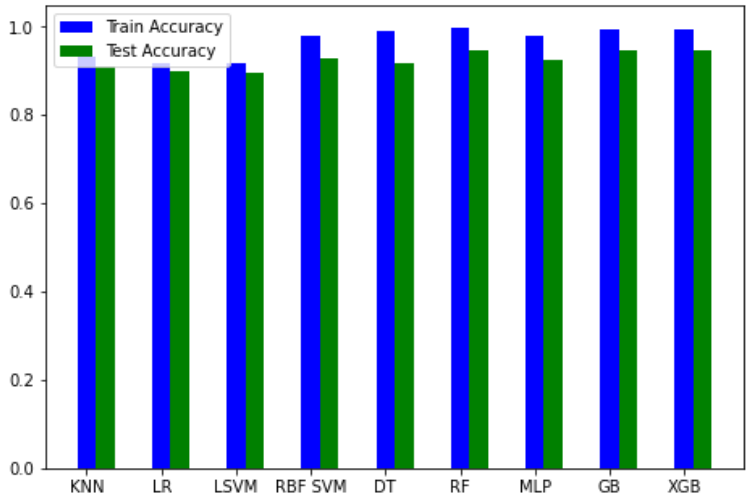
7. Gradient Boosting

For the Gradient Boosting, we did the same experiments: applying the default model without any extra processing regarding the features and by using the parameters returned by the grid search best_params_ method. Also for this classifier, the grid search degrades slightly the performance of the initial model; we passed from 94.6% to 93.8% and from 84.3% to 82.1% for the recall.

8. Extreme Gradient Boosting

For the Extreme Gradient Boosting, we did the same as the last two models. First, when applied to our already scaled data, we got an accuracy of 94.6% and a recall of 84.3%. Then after tuning the parameters by the Grid Search method and using them, the recall improved to 87.1% whereas the accuracy stayed the same at 94.6%. Thus, contrary to the last two models, the Grid Search ameliorated the Extreme Gradient Boosting model for this problem.

Results



Graph 1: Comparison of test and train accuracy obtained for different classifiers

Without using any of the features selection methods and by using the best models for each case as mentioned in the previous section, we notice that the lowest accuracy rates are obtained by the models Logistic Regression et Linear SVM, whereas the highest accuracy is obtained with Random forest. As there is not a big difference on how the models perform for this prediction, achieving our goal became more difficult. Thus, to make a choice and pick the best model we used the features selection methods. We applied the same models and we compared the metrics to choose the best.

	Model	Test Precision	Test Recall	Test f1 Score	Test Accuracy
5	Random Forest	0.889706	0.864286	0.876812	0.946372
7	Gradient Boosting	0.907692	0.842857	0.874074	0.946372
8	Extreme Gradient Boosting	0.876812	0.864286	0.870504	0.943218
3	RBF SVM	0.845588	0.821429	0.833333	0.927445
4	Decision Tree	0.805556	0.828571	0.816901	0.917981
0	K Nearest Neighbors	0.873874	0.692857	0.772908	0.910095
1	Logistic Regression	0.762238	0.778571	0.770318	0.897476
6	MLP	0.756944	0.778571	0.767606	0.895899
2	Linear SVM	0.762590	0.757143	0.759857	0.894322

Table 1: Results for different models without features selection

	Model	Test Precision	Test Recall	Test f1 Score	Test Accuracy
5	Random Forest	0.870504	0.864286	0.867384	0.941640
7	Gradient Boosting	0.847222	0.871429	0.859155	0.936909
8	Extreme Gradient Boosting	0.833333	0.857143	0.845070	0.930599
4	Decision Tree	0.809859	0.821429	0.815603	0.917981
3	RBF SVM	0.789855	0.778571	0.784173	0.905363
6	MLP	0.784173	0.778571	0.781362	0.903785
0	K Nearest Neighbors	0.816667	0.700000	0.753846	0.899054
2	Linear SVM	0.717241	0.742857	0.729825	0.878549
1	Logistic Regression	0.718310	0.728571	0.723404	0.876972

Table 2: Results for different models with features selection (intrinsic method)

We can see that the prediction with features selection performs well even though the performance is not as high as without features selection. The best model is still Random Forest which reached an accuracy of 94.2% and a recall of 86.7% in testing compared to 94.6% and 87.7% respectively in testing without features selection. This is because the dataset used is a public dataset whose features have already been selected and the 11 not selected features still have an influence on the target. However, we were able to find the 10 most important features among the 21, which are:
abnormal_short_term_variability, percentage_of_time_with_abnormal_long_term_variability, prolongued_decelerations, histogram_mean, baseline value, mean_value_of_short_term_variability, histogram_min, accelerations, histogram_median, histogram_width.

Conclusion

We tried various models in order to pick the one with the best performance and by using different features selection methods. In all the experiments, by comparing the metrics, we noticed that the random forest model has the best performance. Without using a feature selection we reached an accuracy of 94.1% for the testing set. However, even by reducing the number of features using the intrinsic method, we get a similar accuracy and recall (values) for this model. We ameliorate the RandomForestClassifier by using a grid search cross-validation which returns the tuned parameters to use to get the highest performance. We assume that this model under the same conditions will give the best performance for this prediction problem. However, to discard some features that are related to sensitive information and problems, especially health, should be approved by a specialized person in this domain, but machine learning can help doctors in their prediction.

Contributions

Amassaghrou Abdelali : Implementing KNN.
Senane Zineb : Helped with feature selection analysis. Implementing Logistic Regression, MLP classifier, Gradient Boosting classifier and Extreme Gradient Boosting Classifier.
Zheng Estelle : Feature selection analysis. Implementing SVM, Decision tree and Random Forest.
All the members contributed to the data analysis part and to the report.

Gitlab link of our repository
https://gitlab.eurecom.fr/zhenge/fetal-risk-prediction

Bibliography
[1] Cardiotocography Data Set - UCI
https://archive.ics.uci.edu/ml/datasets/cardiotocography

[2] Fetal Health Classification - Kaggle
https://www.kaggle.com/andrewmvd/fetal-health-classification

[3] Use of Machine Learning Algorithms for Prediction of Fetal Risk using Cardiotocographic Data - IJABMR
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6822315/pdf/IJABMR-9-226.pdf