Fetal Health Classification from Cardiotocography (CTG) Data

Group 1, Team: Oleksandr Mazur

Project repository: https://github.com/sqcode06/ids-ctg

---

2. Business understanding

Identifying business goals

Background

Cardiotocography (CTG) is routinely used in obstetrics to monitor fetal heart rate patterns and uterine contractions in late pregnancy and labor. Correct interpretation of CTG traces helps clinicians detect fetal distress, but manual visual assessment is subjective and time-consuming. Machine learning models trained on labelled CTG features can support clinicians by providing consistent, quantitative risk estimates for fetal health (normal, suspect, pathological). The dataset for this project is a publicly available fetal health classification dataset derived from CTG exams.

Business goals

The main goal of the project is to build a multi-class prediction model that, given CTG-derived numerical features, classifies fetal health into three categories: normal, suspect, and pathological. The emphasis is not only on overall accuracy but specifically on avoiding underestimation of risk, i.e. reducing cases where a suspect or pathological fetus is predicted as normal. The project is framed as a decision-support tool rather than an autonomous diagnostic system.

Business success criteria

From a “business” perspective, the project is successful if it demonstrates that:

- The model clearly outperforms simple baselines (e.g. majority class, logistic regression) in macro-averaged F1 and balanced performance across all three classes.
- Recall for the pathological class is high (target ≥ 0.90 on held-out data), with a clear explanation of trade-offs between recall and false positives.

- The results are interpretable enough (feature importance, partial dependence or similar) that a clinician could understand which CTG characteristics drive predictions.

Assessing the situation

Inventory of resources

- Public Kaggle dataset "Fetal Health Classification" (2126 samples, 22 numerical features + target).
- Computing environment: personal laptop with Python 3, Jupyter, Git, and common libraries (pandas, NumPy, scikit-learn, XGBoost, matplotlib).
- Time: approximately 60 hours of individual work until the final project deadline.
- Supervision: access to course TAs and lecture material in Introduction to Data Science.

Requirements, assumptions, and constraints

- The project is purely educational; no real-time clinical deployment is planned.
- Only the provided tabular dataset is used; no external patient data will be collected.
- All features are numeric and already pre-engineered, so feature engineering is mostly limited to transformations and selection.
- The project must fit within the scope and level of an introductory data science course, with reasonable model complexity and clear reporting.

Risks and contingencies

- The dataset is imbalanced (normal ≫ suspect, pathological), which may lead to poor performance on the minority classes if not handled properly.
- Time risk: implementing many advanced models may exceed available time; contingency is to focus on a small set of well-tuned tree-based models.
- Overfitting risk: high-capacity models (e.g. boosted trees, XGBoost) might memorize the dataset; mitigation: stratified cross-validation and careful performance reporting.

Terminology

- CTG – cardiotocography;
- Normal / suspect / pathological – fetal health categories assigned by experts based on CTG;
- False negative – a case where the true class is suspect or pathological but predicted as normal or suspect, respectively.

Costs and benefits

The main cost is the student's time (≈60 hours) and some compute time, which is negligible. The immediate benefit is educational: practising the full CRISP-DM cycle on a medically relevant, imbalanced classification task. In a broader context, the project could contribute a documented, reproducible baseline that future work could extend towards real clinical decision-support.

Defining data-mining goals

Data-mining goals

- Build and compare several classification models (baselines, tree ensembles, XGBoost) for three-class fetal health prediction.
- Explicitly study class imbalance handling and threshold tuning to reduce underestimation of risk, especially for pathological cases.
- Provide interpretable outputs (feature importance, partial dependence, confusion matrices) to understand decision behavior.

Data-mining success criteria

- Macro F1 on the test set substantially higher than a majority-class dummy classifier and simple baselines.
- High recall for the pathological class, quantified and discussed in the context of increased false positives.
- Clear, well-documented pipeline (notebooks + code) and a concise final report/poster describing methods, results, and limitations.

---

3. Data understanding

Gathering data

Outline data requirements

To address the project goals, the required data must:

- Contain individual CTG exam summaries with numeric features describing fetal heart rate, uterine contractions, and related signal characteristics.
- Include a categorical target variable with three expert-assigned labels: normal (1), suspect (2), pathological (3).
- Have enough samples per class to train and evaluate multi-class models (at least a few hundred examples in the majority class and some tens in the minority class).

Verifying data availability

The Kaggle dataset "Fetal Health Classification" satisfies these requirements: it contains 2126 rows and 22 columns, with all 22 being numerical CTG-derived attributes and one target column — `fetal_health`. The data is downloadable as a single CSV file and is suitable for educational machine-learning projects. The dataset license allows non-commercial research and teaching use.

Define selection criteria

The initial plan is to use all records and all feature columns. No filtering of rows is applied beyond possible removal of obvious duplicates if identified. Feature selection, if any, will be done later based on correlation analysis, model feature importance, and domain interpretability. For evaluation, the data will be split into training and test sets using a stratified split to preserve class proportions.

Describing data

Preliminary inspection (via `info()` and `describe()`) shows:

- 2126 non-null entries for all 22 numeric predictors and the target.
- Feature ranges vary: some features (e.g. baseline value) are on a larger scale (≈100–200), others are small proportions or counts (e.g. accelerations, decelerations, histogram statistics).
- No missing values are reported, so classical imputation is not required.

The target distribution is imbalanced:

- `fetal_health = 1.0` (normal): 1655 samples
- `fetal_health = 2.0` (suspect): 295 samples
- `fetal_health = 3.0` (pathological): 176 samples

This imbalance motivates careful choice of metrics (macro F1, per-class recall) and techniques (class weights, threshold adjustment, or resampling).

Exploring data

Basic exploration steps include:

- **Univariate distributions**: histograms and density plots for key features (baseline heart rate, accelerations, uterine contractions, short-term and long-term variability measures) to understand typical value ranges and skewness.
- **By-class plots**: plotting feature distributions separately for each fetal health class to see whether patterns differ between normal, suspect, and pathological groups.

- **Correlation analysis**: computing a correlation matrix for all numeric features to identify strongly correlated or redundant variables (e.g. different histogram statistics of the same underlying CTG signal).
- **Class balance plots**: bar plots of class count to visualize the imbalance and motivate later techniques for handling it.

These steps, apart from general acquaintance with the data, aim to ensure that the meaning and scale of each feature is understood and to detect any obvious data issues, such as extreme outliers and implausible ranges.

Verifying data quality

Several checks have been planned and partly performed already:

- **Missing values and types**: `df.info()` confirms that all predictor columns are non-null float values; no categorical encoding is needed.
- **Duplicates**: the dataset will be examined for exact duplicate rows; if any are found, they will be removed to avoid leakage between train and test sets.
- **Outliers and plausibility**: boxplots or quantile ranges are inspected for major features (e.g. baseline heart rate); values far outside physiologically plausible ranges will be investigated.
- **Class label consistency**: only values `{1, 2, 3}` appear in `fetal_health`; they will be mapped to integer codes `{0, 1, 2}` internally for modelling while preserving the original semantic labels in all reports and plots.
- **Train/test split check**: after stratified splitting, class proportions in train and test subsets are compared to ensure that small classes (suspect, pathological) remain represented for evaluation.

As a result of these steps, the dataset is considered of sufficient quality for a student-level predictive modelling project. Further data preparation (e.g. scaling for linear models, optional resampling, and removal of highly correlated features) will be performed in the data preparation phase of CRISP-DM.

---

4. Planning the project

Project work plan and hours

Planned total effort: 60 hours (single team member: Oleksandr Mazur).

#	Task	Hours
1	Background reading on CTG, fetal health and prior ML work; write short context section	6 h
2	Initial data inspection and basic EDA (class balance, summary statistics, simple plots)	5 h
3	Deeper EDA: correlations, class-wise feature distributions, optional clustering	3 h
4	Baseline models (dummy, logistic regression, decision tree) + initial evaluation	4 h
5	Class imbalance handling (class weights, metric choice, possibly simple resampling)	4 h
6	Study and select advanced models (tree ensembles, XGBoost)	7 h
7	Implement and tune advanced models using cross-validation and small hyperparameter grids	8 h
8	Model interpretability and error analysis (feature importance, partial dependence, confusion matrices)	8 h
9	Prepare final notebook(s), written report, and project poster	10 h
10	Administrative tasks, repository maintenance, reruns, and small fixes	5 h

Methods and tools

Planned tools and methods:

- Python, Jupyter notebooks, Git/GitHub;
- **Libraries**: pandas, NumPy, scikit-learn, XGBoost, matplotlib, seaborn;
- **Methods**: stratified train/test split, cross-validation, baseline classifiers, tree-based ensembles (HistGradientBoosting, XGBoost), class-imbalance handling, threshold tuning for higher recall of pathological cases, and model interpretability techniques (feature importance, partial dependence, confusion matrices).

Thank you!