

# Project Charter


Mid-Term  
Exercise  
Grade: 90

## Business background

Who is the client, what business domain the client is in.

- German Credit Company, consumer credit risk assessment
- Portuguese Banking Company, marketing and selling banking products


What business problems are we trying to address?

- The companies would like us to create a generalized algorithm that will classify its customers as a good or bad credit risk (German Credit Company) or likely to purchase a certain product (Portuguese Banking Company) 

## Scope

Our project is to build an automatic pipeline that will improve the performance of a given XGBoost model. We will do this, by identifying where performance is weak and improving performance in those areas of the data.

The identification process will consist of an automated splitting function where slices will be combinations of different features in the dataset. These slices will then be judged on a benchmark accuracy measure to determine weakness and the training set will be updated to improve performance in the weak slices.

This solution will be implemented as a python file to be added as a stage after initial model execution. We will generate a new dataset and set of problematic slices. We will then output a new model based on the augmented dataset that will better perform in problem areas. We will also provide in the output the list of problematic slices and by how much performance increased in those subsets. 

## Personnel

Dream Team: 

- Tomer
- Esther
- Eli
- Ziv

# Metrics

The goal is to reduce the number of bad credit risks taken by the bank, by predicting ahead of time which customers will subscribe to a term deposit or are bad/good credit risks. At the end of this analysis pipeline, problematic slices of the test data will be extracted with their corresponding accuracy metrics.

We can use the overall data accuracy metrics and the slice's accuracy metrics as the goal we aim to improve with augmented data. We do not know the slices yet, but we know the current overall accuracy metrics over the whole Bank marketing dataset:

Bank marketing dataset	Accuracy	Recall	Precision	F1 Score
Label 1 (yes)	0.90408434090 23887	0.37366099558 916194	0.658888888888 88889	0.47687977482 911137
Label 0 (no)	0.90408434090 23887	0.97436753778 07465	0.92150979153 506	0.94720181810 80314

Table 1: Bank marketing dataset baseline

In addition, the German credit risk dataset model's initial accuracy metrics are:

German credit risk dataset	Accuracy	Recall	Precision	F1 Score
Label 1 (good)	0.75	0.31	0.67	0.42
Label 0 (bad)	0.75	0.94	0.76	0.84

Table 2: German credit risk dataset baseline

The Recall score of the datasets are typical for unbalanced datasets:



	Count Label 1	Count Label 0
Bank marketing dataset	39922	5289
German credit risk dataset	700	300

Table 3: Bank marketing dataset unbalance

As seen in (Table 3), the bank marketing dataset is unbalanced with a ratio of 1 client subscribed to about 7.5 clients who did not subscribe. Since we are focusing on mitigating the imbalance effect, we will use F-score as our main accuracy measure.

# Plan

Phases (milestones), timeline, short description of what we'll do in each phase.

1. Run the model with the preprocessed data to establish baseline performance.
2. Implement the data slicing algorithm on our test set to identify which slices of data are underperforming.
  - a. Split test set into slices
  - b. Determine which slices are problematic
3. Generate additional synthetic data similar or identical to the problematic data slices and retrain the model with the new dataset to improve its accuracy on those problematic slices.
4. Compare model performance on problematic slices before and after updating the training dataset.

## Architecture

### Data

We will be using two datasets from the UCI Machine Learning Repository:

#### **1. Bank Marketing**

Dataset:

- 7 numeric features
- 9 categorical features
- 1 binary label
- 45211 samples

Libraries:

- Numpy
- Pandas
- Sklearn
- Matplot

Feature manipulations:

- 7 numeric features stay the same
- 9 categorical features become one-hot encoded into 45 features
- The binary label is replaced with (0/1)
- The samples are split into two sets of [0.7,0.3] with stratify

#### **2. Statlog (German Credit Data)**

Dataset:

- 7 numeric features
- 13 categorical features
- 1 binary label

- 1000 samples

Libraries:

- Numpy
- Pandas
- Sklearn
- Seaborn

Feature manipulations:

- 7 numeric features are standardized
- 13 categorical features become one-hot encoded into 55 features
- The binary label is replaced with (0/1)
- The samples are split into two sets of [0.8, 0.2] without stratify

## Creating Slices

Once the model is trained and tested, we can begin to examine which 'slices' (subsets of data) are underperforming based on our accuracy metrics. Below are a few issues we may face with creating slices and how we will address them.

For each additional feature of  $k$  values being considered in slices, the number of slices increases  $k$  times. If we consider all qualitative features in the German credit dataset, this would lead to 59,400,000 unique slices. We can address this issue by keeping the slices univariate and bivariate. We will only look at combinations of up to two features per slice to convey low order interaction and keep the slices explainable.

### **Numeric features slices - Highest Density Regions (HDR):**

The first method we would use to slice the dataset is to describe the density (axis  $y$ ) as a function of a numeric feature (axis  $x$ ). After finding the smallest single interval region where a proportion of the data is found, we iteratively shrink that area by some epsilon and check if the performance on the new region is more or less accurate. This way we can find an area for improvement for a numeric feature. This method retrieves a univariate slice of a single interval (and not a union of intervals) to keep computation low, avoid overfitting and stay explainable.

### **Optional augmentation 1 - adding numeric intervals from HDR to decision trees:**

Since in the HDR method we find a single interval region per numeric feature, we can turn this into a binary categorical feature of {numeric value} in feature region or out of feature region for each sample, and add this to the decision trees. This way we can find bivariate slices including numeric features.




### **Categorical features slices - Decision Trees:**

The second method we will use is to create decision trees and look to find what splits of those features minimize our accuracy. Like previously, we will limit the combinations of features to up to two, for the same reasons, giving us explainable interaction areas of poor performance.


### **\*\*Optional\*\* augmentation 2 - slices containing an unwanted misclassification:**

It is mentioned in the German credit risk dataset that: “It is worse to class a customer as good when they are bad, than it is to class a customer as bad when they are good.”. Therefore, false positives are less harmful than false negatives.


We can augment the decision tree so that it prioritizes finding false negatives over false positives if requested. 

## Determining the Most Problematic Slices

We will use the following measures to determine if slices are problematic are:

- **Minimal support:** We will return slices that pass a threshold number/percentile of error. 
- **Statistically significant:** We will look for slices with significant drops in performance.

## Updating Dataset Based on Problematic Slices

Once we identify the problematic slices, we will want to increase their prevalence in the dataset to help improve the model training. We can do this by adding similar or identical samples from the problematic slice. One method for creating ‘similar’ data is to “mutate” numeric features slightly to add some generalization.  We can repeat this for each problematic slice and then retrain the model on our augmented dataset.

## Communication

We will keep in touch by whatsapp and have weekly Zoom meetings.

## Project Resources

- <https://towardsdatascience.com/how-to-find-weaknesses-in-your-machine-learning-models-ae8bd18880a3A>
  - [Pdf version](#)
- <https://arxiv.org/pdf/1807.06068.pdf>
- <https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/>
- <https://arxiv.org/pdf/2108.05620.pdf> - FreaAI: Automated extraction of data slices to test machine learning models