# Oversampling methods introduction

1. Oversampling methods

* Process of increasing number of samples from the minority class
* until a desired balancing ratio

1. Random extraction vs sample generation

* Sample extraction: extract samples at random from minority class
* Random
* Sample generation: creates new samples from existing ones
* SMOTE and its variants
* ADASYN

A close-up of a sample

Description automatically generated

# Random oversampling

1. Overview

* Extracts obs at random from the minority class until a certain balancing ratio is reached
* Naïve technique

1. Duplication

* Extracts observations at random from the minority class, with replacement
* Duplicates samples from the minority class
* Increase likelihood of overfitting

1. Implementation

A black text on a white background

Description automatically generated

1. Multiclass

* 1 minority class: sampling\_strategy = ‘auto’
* Multiple minority classes: sampling\_strategy = ‘not majority’ or dict

Random oversampling with smoothing

1. Overview

* Extracts obs at random from the minority class and compounds its value with some noise
* The noise is informed by the class distribution
* Creates new examples -> avoids data duplication
* We can choose how disperse we want the new samples -> shrinkage factor (arbitrary)

1. Procedure

* Take the minority samples and determine their distribution -> Std of each variable
* Extract a value at random from N ~(0, 1)
* Extract at random 1 observation
* Add the noise (with the shrinking factor) = shrinking factor x std x random value

1. Implementation

A screenshot of a computer code

Description automatically generated

A graph of a number of dots

Description automatically generated with medium confidence

# SMOTE

1. Overview

* Synthetic Minority Oversampling Technique
* Creates samples by interpolation – a type of estimation where new data points are created within the range of known data points
* Minority class is oversampled by creating synthetic examples instead of extracting data at random
* Prevents duplication

1. How it works

* Looks only at the obs from the minority class
* Finds its k nearest neighbours (typically k=5)
* Determines the distance between the neighbours and the sample we want to generate a new observation from
* Multiplies that distance by a random number and adds it to the original sample to place the new obs in the dataset

A diagram of a star formation

Description automatically generated

1. Numerical example
2. Python implementation

* Isolates minority class samples
* Trains KNN and finds K nearest neighbours to each sample of the minority class
* Determines how many new samples need to be generated
* Selects from which samples a new sample will be generated (random)
* Selects the neighbour that will be used to extrapolate the sample (random)
* Finds a random factor
* Determines new sample value

1. Implementation

A close-up of a math problem

Description automatically generated

# SMOTE-NC

1. Overview

* SMOTE – Nominal Continuous
* Extends functionality of SMOTE to categorical variables
* SMOTE, its variants and ADASYN can’t work with categorical data (unlike random oversampling)

1. How it works

* Calculate the Euclidean distances to find K neighbours
* Calculate standard deviations of numerical variables in minority class -> median of standard deviations. When calculating L2 distance between 2 obs.
* If values in categorical var are the same -> 0
* If values in categorical var are different -> use the median stddev.
* Values of numerical variables are calculated as in SMOTE
* Values of categorical variables are those shown by the majority of neighbours

1. Implementation

A screenshot of a computer code

Description automatically generated