FLATIRON SCHOOL

Affirming Structural Integrity with Data Preprocessing Techniques.

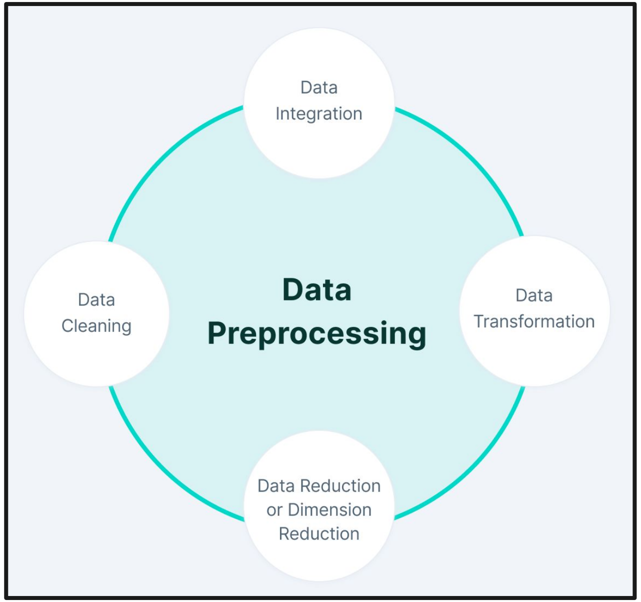
DS

hosted by KASH

CRITICAL QUESTION

How can we optimize the structural integrity and heuristic accessibility of our dataset through advanced processing techniques?

THE MAJOR ELEMENTS OF DATA PREPROCESSING



Data preprocessing generally comprises more advanced methodologies for **cleaning and restructuring our data in line with easier machine learning and modeling**.

As such, the techniques within this umbrella include more mathematically and operationally exhaustive processes spanning the range of **cleaning, integration, transformation, and reduction** of data.

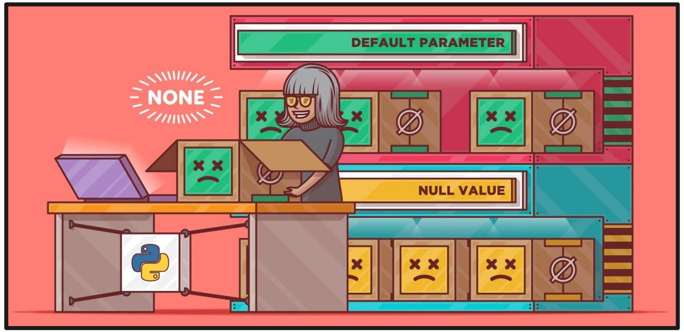
WRITTEN IDEATION

WHAT METHODS CAN YOU USE TO **FURTHER PROCESS YOUR DATA** *AFTER* HAVING PERFORMED **BASIC MODELING**?

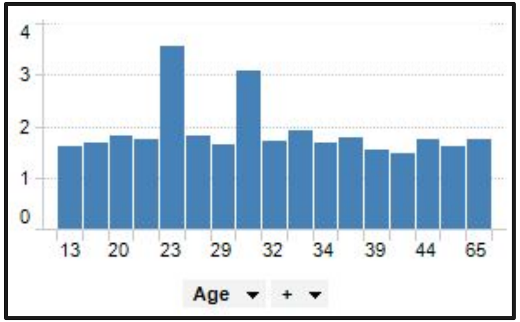
DATA CLEANING: NULL VALUE IMPUTATION

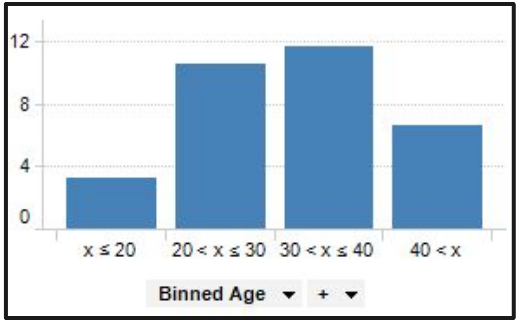
A classic tool in our arsenal is that of **null value imputation**, which states that null values throughout our dataset can be *empirically removed and outcast in order to highlight the signal-carrying data*.

However, imputation encompasses other techniques for handling null values, such as **mean/median/mode replacement**, **proximal observation carrying**, and even **nearest-neighbor classification imputation**.



DATA CLEANING: NOISE MANIPULATION





The full range of data within a dataset can often comprise **too weak of a signal-to-noise ratio**, despite the prevalence of useful signal heuristic to simply drop or get rid of.

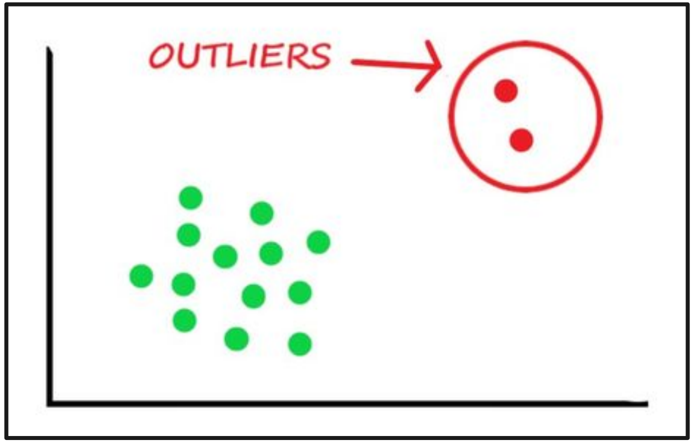
As such, techniques such as **binning and clustering data** from continuous/sparse ranges into more compartmentalized domains is highly popular for **retaining signal while losing unimportant variability and noise**.

This technique *can* backfire however due to some **signal loss**.

DATA CLEANING: OUTLIER DETECTION

Outliers can be heavily detrimental to the accuracy and reliability of a predictive model due to their ability to **numerically skew the patterns and heuristics that the algorithm is attempting to learn**.

Similarly to null values, while there are many techniques for imputing outliers, oftentimes the best tactic is simply **dropping or cutting them entirely** from our data using techniques such as **Tukey's Method**.



RECOMMENDED RESOURCE

A Brief Overview of Outlier Detection and Removal Techniques for M.L.

DATA INTEGRATION: DATA DICTIONARY CURATION

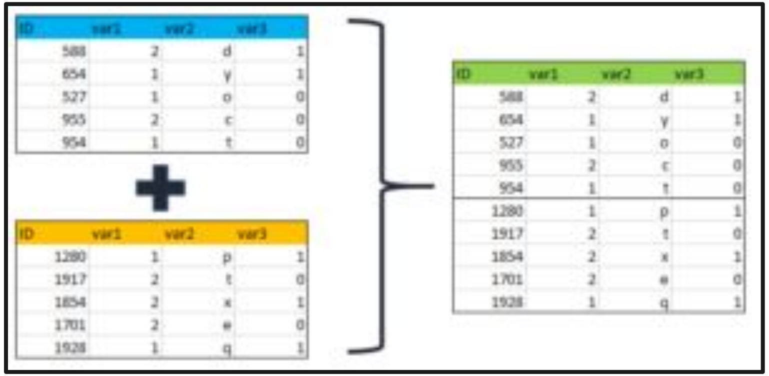
Data dictionaries are rarely provided with datasets – instead, it's often highly appropriate to **curate one from scratch through domain research and data source investigation** in order to best understand what information our data, features, and domain actually provide and what relevant findings we can interpret.

Column name	Definition	Data type	Required
Name	This column refers to the first name of customers	String	Yes

DATA INTEGRATION: DATASET-TO-DATASET ASSOCIATION

In some projects, there can be highly useful signal spread out across multiple datasets (in some cases, from entirely different sources) – being able to **integrate datasets using similar features, indices, and other references** within the data is extremely useful and can make a high-level data investigation much, much easier.

Languages like **SQL** and **R** excel at this particular skill in addition to Python.



ID	var1	var2	var3
588	2	d	1
654	1	y	1
527	1	o	0
955	2	c	0
954	1	t	0

ID	var1	var2	var3
1280	1	p	1
1917	2	t	0
1854	2	x	1
1701	2	e	0
1928	1	q	1

ID	var1	var2	var3
588	2	d	1
654	1	y	1
527	1	o	0
955	2	c	0
954	1	t	0
1280	1	p	1
1917	2	t	0
1854	2	x	1
1701	2	e	0
1928	1	q	1

DATA INTEGRATION: METADATA TRACKING

As we start to combine processing, preprocessing, and exploration/visualization methods together throughout a full-scale data pipeline, the **sheer number of changes and alterations made to the dataset** can mandate some method of tracking said changes.

By designing our datasets in an **object-oriented manner that can support substates and inheritance of relevant metrics**, we can explicitly track metadata on our dataset's journey from ingestion to prediction.

RECOMMENDED RESOURCE

Practice with Data Aggregation and Grouping using Pandas

DATA TRANSFORMATION: STANDARD SCALING

The scale of relevant features can make a massive impact on how well our model is able to learn patterns – as such, sometimes it is relevant to standardize scales of all features such that **each feature's comprised data falls within the same range as every other feature**.

A reliable tool to achieve this result is the **StandardScaler()** tool within Scikit-Learn.

$$z = \frac{x - \mu}{\sigma}$$

```
from sklearn.preprocessing import StandardScaler
```

```
sta = StandardScaler()
```

DATA TRANSFORMATION: FEATURE NORMALIZATION

In other cases, more dramatic changes may be warranted to ensure generality and comparability across different features in order to dilute outlier/skew effects or sample more appropriately – in these cases, data normalization is a handy technique to have.

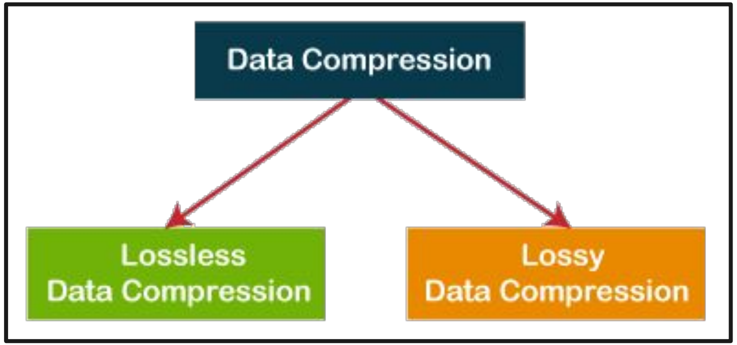
A reliable tool to achieve this result is the **Normalizer()** tool within Scikit-Learn.

```
In [83]: from sklearn import preprocessing
         import numpy as np

In [84]: numpy_array = np.array([2,3,5,6,7,4,8,7,6,17,18,19,2,1,89])

In [85]: normalized_array = preprocessing.normalize([numpy_array])

In [86]: print(normalized_array)

[[0.02086505 0.03129758 0.05216263 0.06259516 0.07302769 0.04173011
  0.08346021 0.07302769 0.06259516 0.17735295 0.18778548 0.19821801
  0.02086505 0.01043253 0.92849488]]
```

RECOMMENDED RESOURCE

Using Standardization and Min-Max Optimization to Transform Data

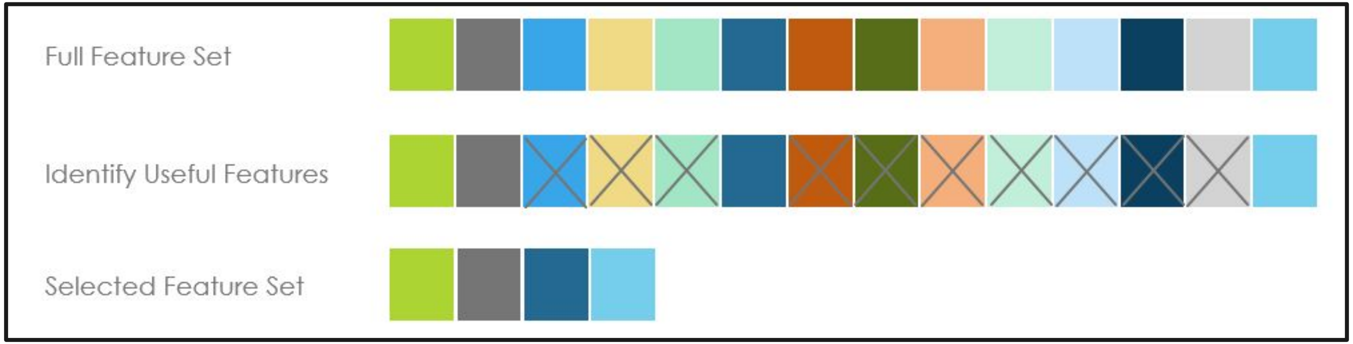
DATA REDUCTION: VALUE COMPRESSION



With more advanced and large-scale datasets, assessing the impact of the size of our data can be important, as the **performance and runtime of our algorithms are *both* impacted by our data size**.

As such, we can perform **compression techniques to reduce our data size while attempting to retain as much signal as possible** – in these cases, it's important to denote whether our compression methods are **lossy** or **lossless** for our data.

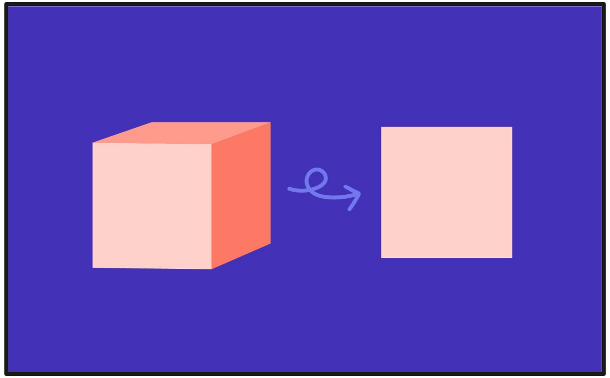
DATA REDUCTION: FEATURE SELECTION



Not all features contribute equally and reliably to our data's predictive capacity as well – **some features are better to be discarded and cut out entirely** rather than be retained for modeling.

Most **basic feature selection techniques are very manual and mandate direct observation** prior to dropping desired features.

DATA REDUCTION: DIMENSIONALITY REDUCTION



In other cases, it's not as simple – **some features may simply compose low-level or high-level signal** and it's not as easy to just drop some and retain others.

In these cases, we have to rely on more advanced linear-algebra-related techniques to **convert larger sets of features into smaller sets of features** through a method called **"dimensionality reduction"** (sort of like using multiple ingredients to make a tasty soup).

RECOMMENDED RESOURCE

Technical Tutorial for Implementing Principal Component Analysis for M.L.

WRITTEN IDEATION

WHICH **SPECIFIC TECHNIQUES** WITHIN THE REALM OF **DATA PREPROCESSING** DO YOU WANT TO EXPERIMENT WITH FIRST?

THANK YOU FOR YOUR TIME!