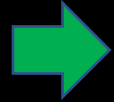


Pattern Recognition

Data Preprocessing

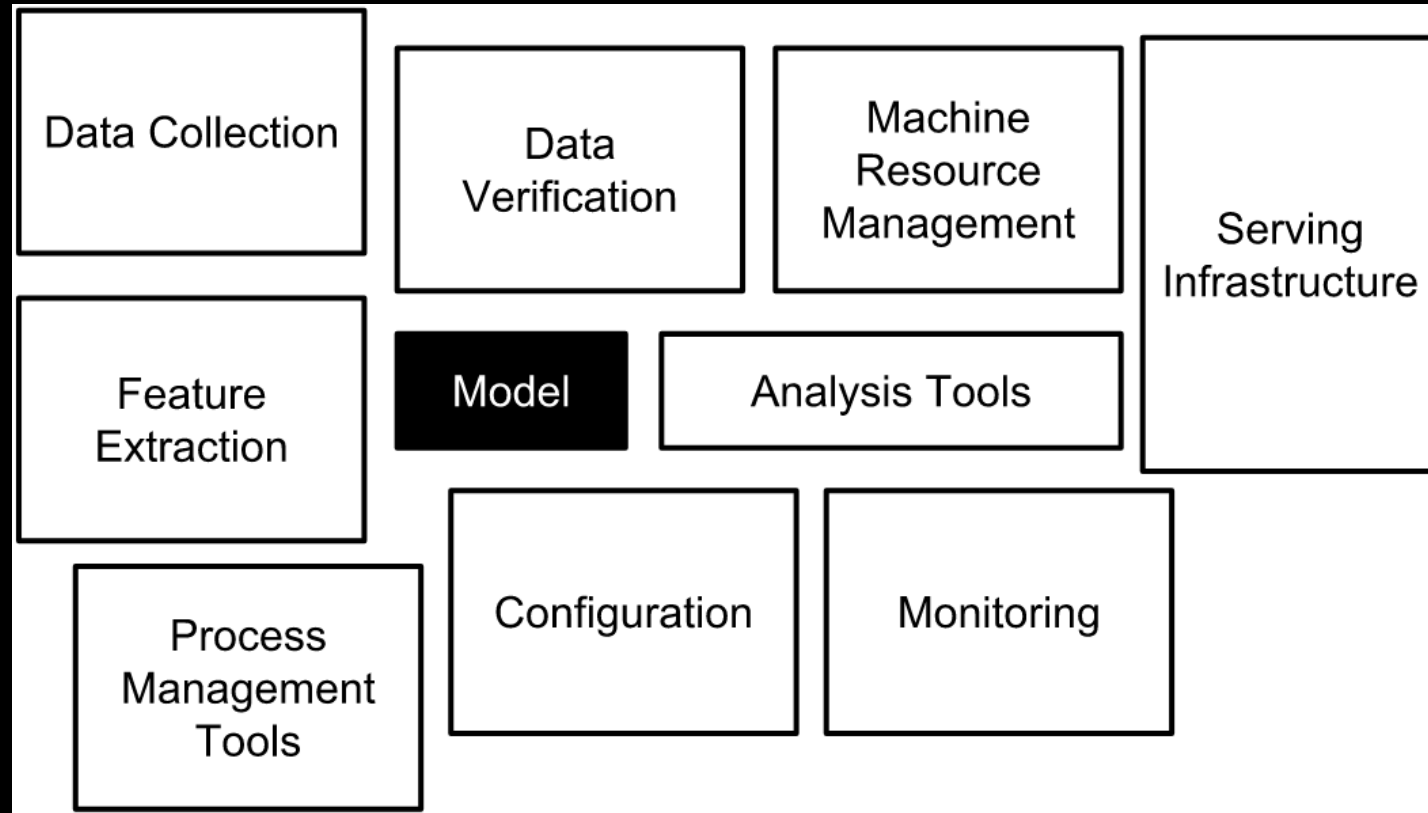
Content



Content
Introduction
Preprocessing Operations
Data Preprocessing techniques

Introduction

- Building an ML model is a multistep process.
- Each step presents its own technical and conceptual challenges.

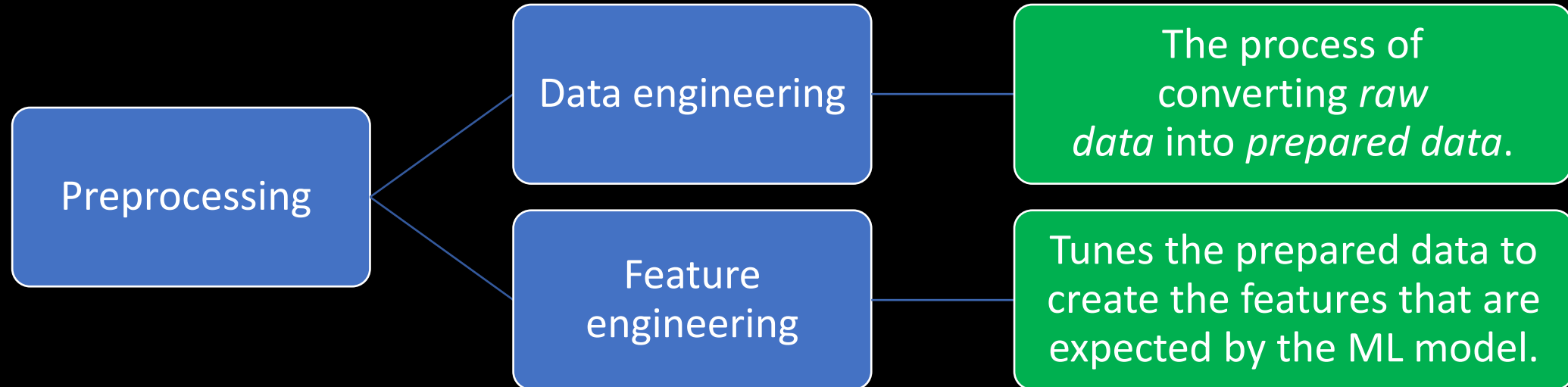


Introduction

- The size of training datasets in real-world can be **equal to or greater than one terabyte (TB)**.
- Therefore, you need **large-scale data processing frameworks** in order to process these datasets efficiently and distributedly.
- When you use an ML model to make predictions, you must **apply the same transformations** that you used for the training data on the new data points.
- By applying the same transformations, **you present the live dataset to the ML model the way that the model expects.**

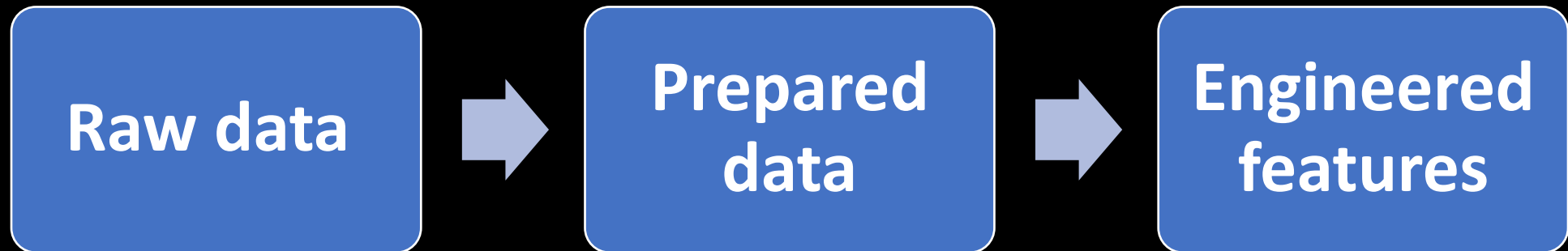
Introduction

- Preprocessing the data for ML involves both data engineering and feature engineering.



Introduction

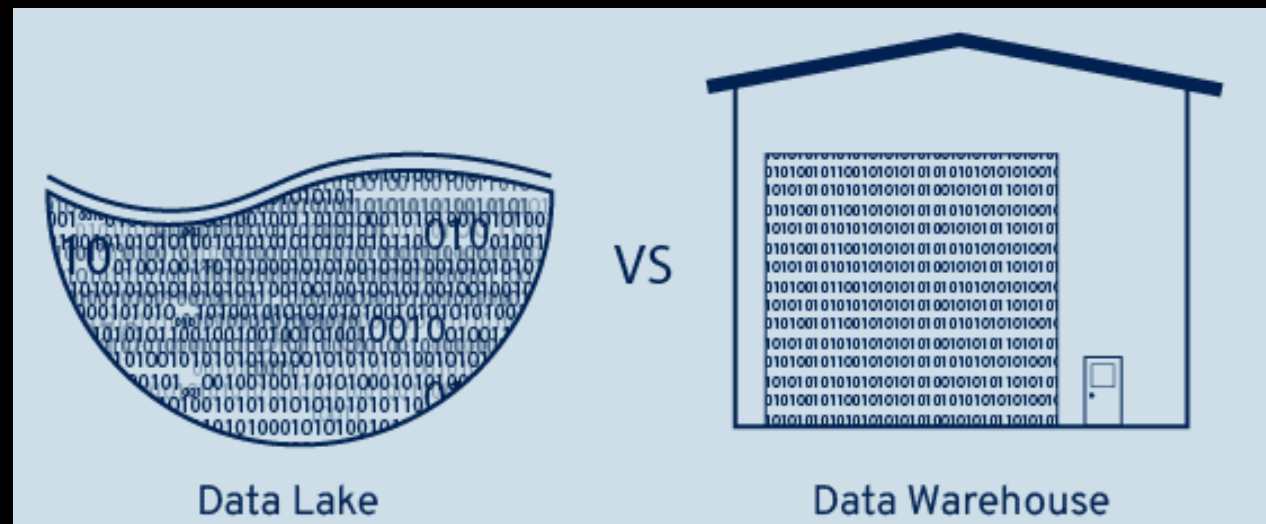
- Data can be presented in different forms:



Introduction



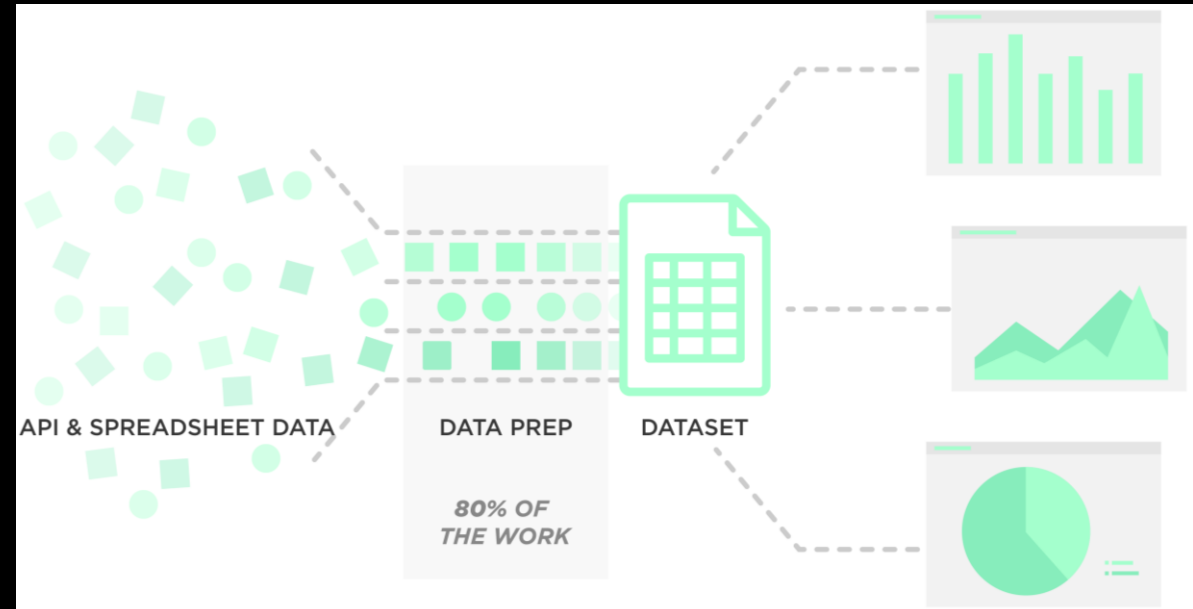
- The data in its source form, without any prior preparation for ML.
- The data might be in its raw form (in a data lake) or in a transformed form (in a data warehouse).
- **Raw data** means that the data hasn't been prepared specifically for your ML task.



Introduction



- The dataset in the form ready for your ML task.
 - For example, each row in the dataset represents a unique customer, and each column represents the attributes of the customer.
- In a prepared data table,
 - irrelevant columns have been dropped,
 - invalid records have been filtered out.
 - For supervised learning tasks, the target feature is present.

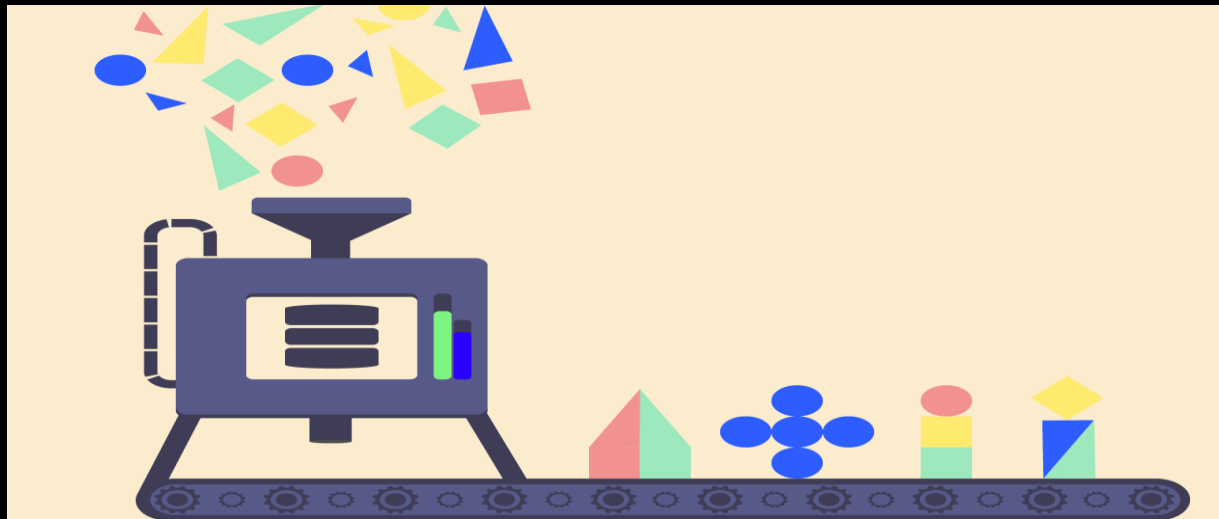


Introduction



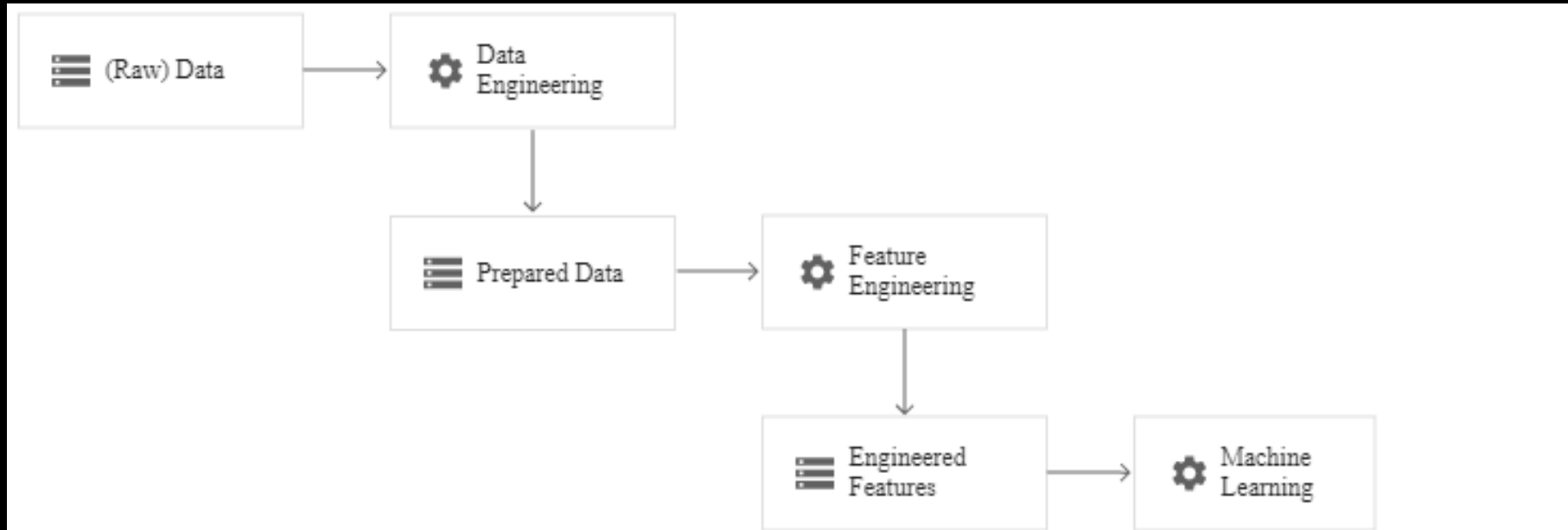
- The dataset with the tuned features that are expected by the model.
- Features are created by performing certain ML-specific operations on the columns in the prepared dataset.

<https://cdn.analyticsvidhya.com/wp-content/uploads/2021/05/64590automated-feature-engineering.png>

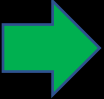


Introduction

- Steps that are involved in preparing preprocessed data



Content

	Content
	Introduction
	Preprocessing Operations
	Data Preprocessing techniques

Preprocessing Operations

- For structured data, data preprocessing operations include the following:

Data cleansing

- **Removing or correcting records** that have corrupted or invalid values from raw data and removing records that are missing many columns.

Instances selection and partitioning

- Selecting data points from the input dataset to create training, evaluation (validation), and test sets.
- This process includes techniques for **repeatable random sampling**, **minority classes oversampling**, and **stratified partitioning**.

Feature tuning

- Improving the quality of a feature for ML.
- Includes **scaling** and **normalizing numeric values**, **imputing missing values**, **clipping outliers**, and **adjusting values that have skewed distributions**.

Preprocessing Operations

Feature transformation

- Converting a numeric feature to a categorical feature (**bucketization**)
- Converting categorical features to a numeric representation (**one-hot encoding, learning with counts, sparse feature embeddings**, etc.)

Feature extraction

- Reducing the number of features by creating lower-dimension, more powerful data representations using techniques such as **PCA, embedding extraction**, and **hashing**

Feature selection

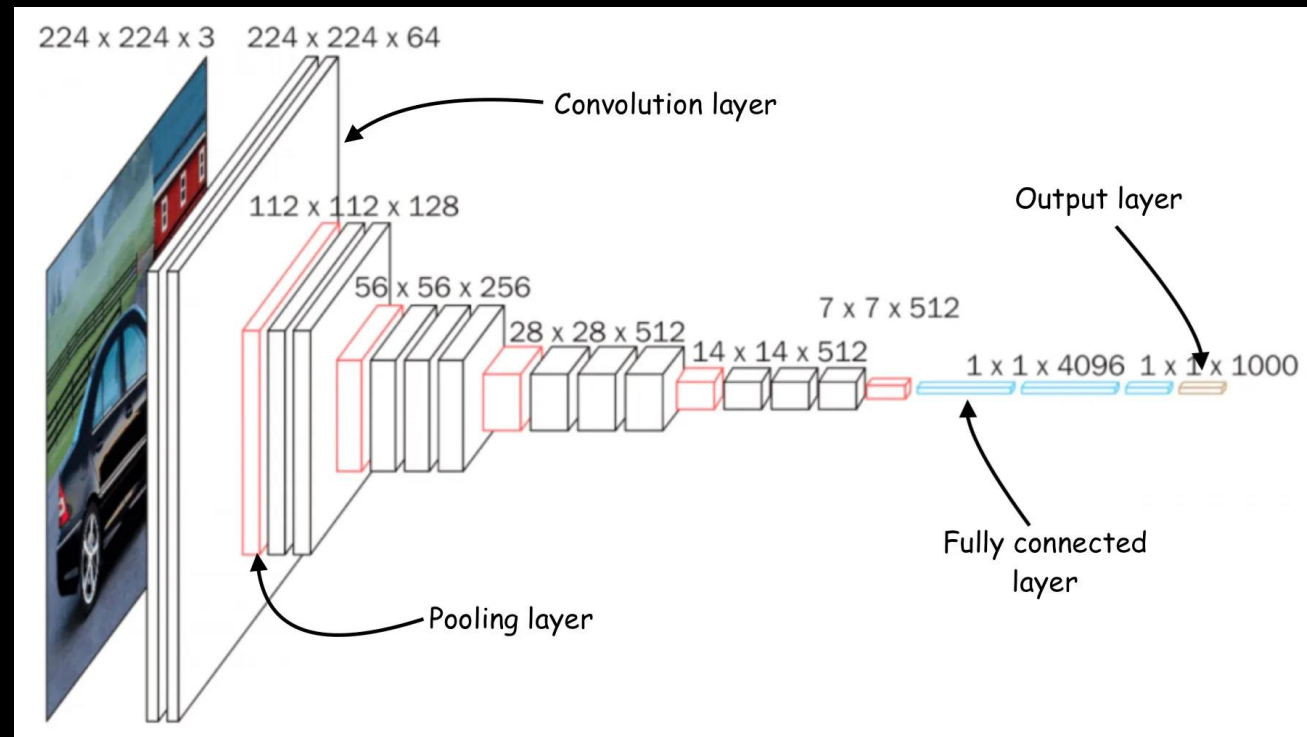
- Selecting a subset of the input features for training the model, and ignoring the irrelevant or redundant ones, using **filter methods**.
- Feature selection can also involve simply dropping features if the features are missing many values.

Feature construction

- Creating new features by using techniques, such as **polynomial expansion** or **feature crossing**.
- Features can also be constructed by using business logic from the domain of the ML use case.

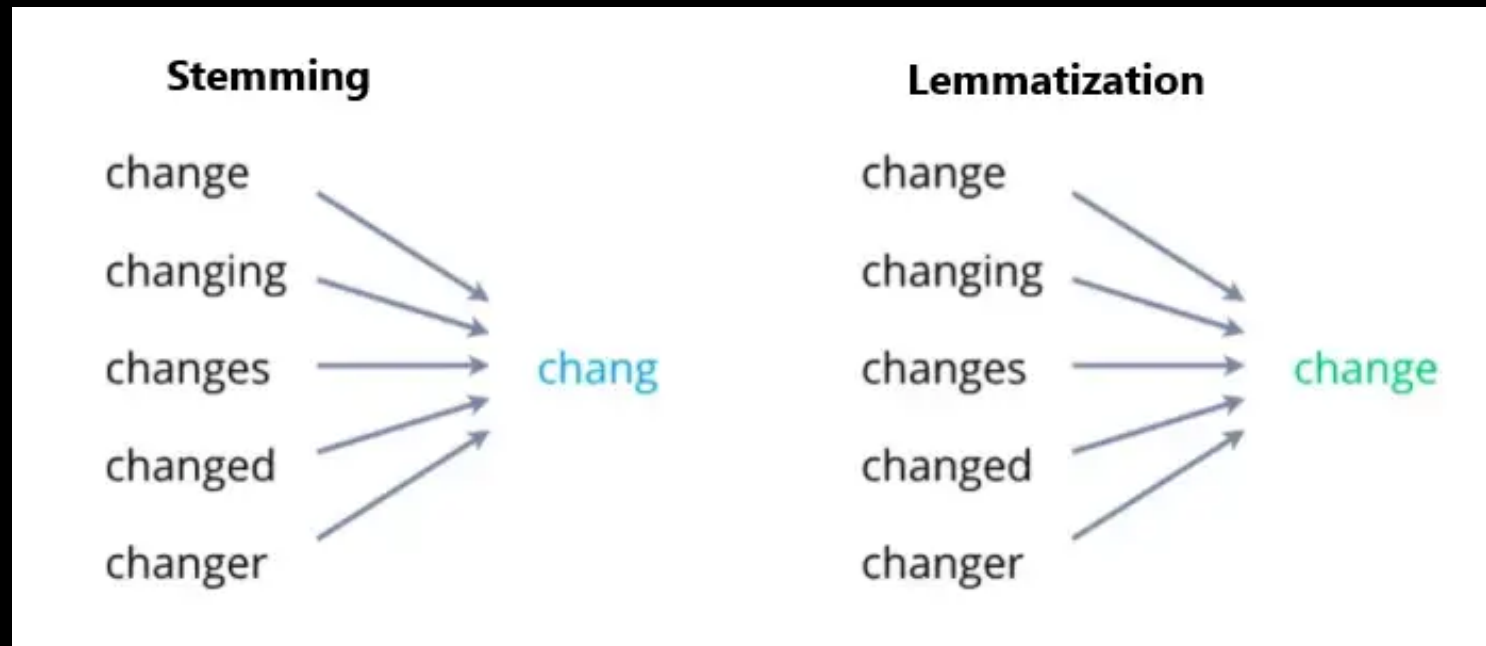
Preprocessing Operations

- For unstructured data (images, audio, or text documents), deep learning architectures do feature engineering.
- Constructing the right architecture requires some empirical knowledge of the data.



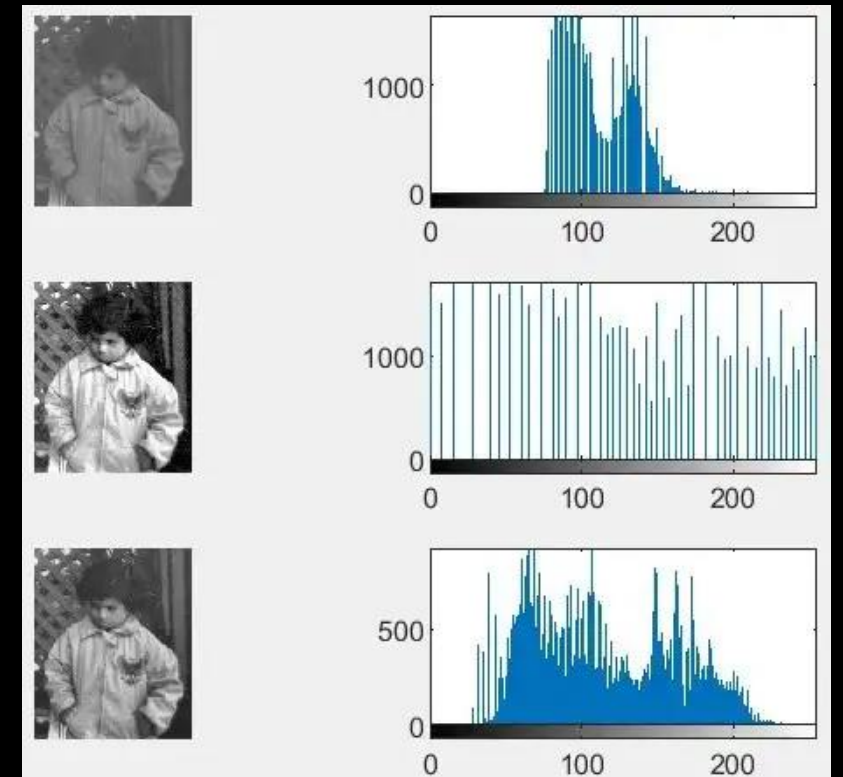
Preprocessing Operations

- For unstructured data some amount of preprocessing is needed, such as the following:
 - **For text documents:** stemming and lemmatization, TF-IDF calculation, and n-gram extraction, embedding lookup.



Preprocessing Operations

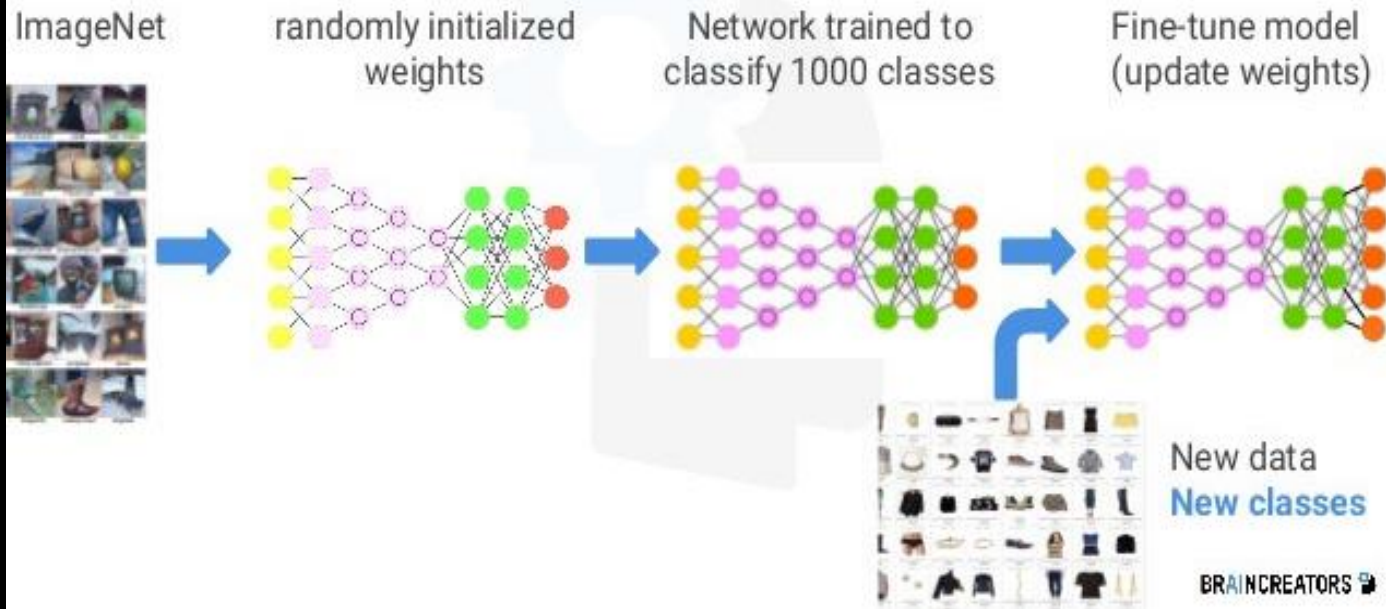
- For unstructured data some amount of preprocessing is needed, such as the following:
 - **For images:** clipping, resizing, cropping, Gaussian blur, and canary filters.



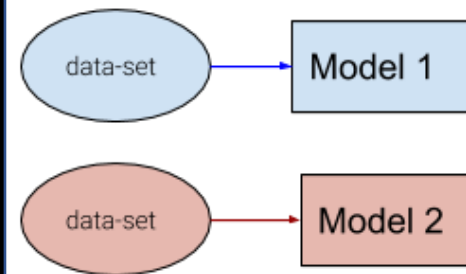
Preprocessing Operations

- Machine learning models can transfer information from one task to another.
- This is known as **transfer learning**.

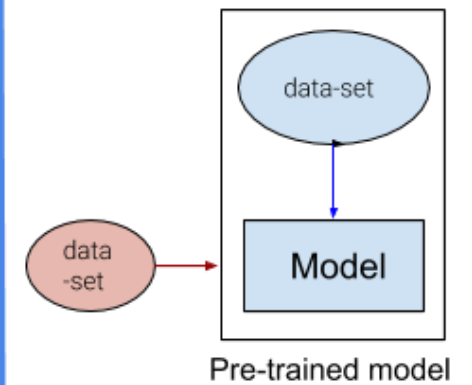
Transfer Learning



Without Transfer Learning



With Transfer Learning



Preprocessing Operations

- Preprocessing and transformation operations can be categorized as follows

Instance-level transformations during training and prediction.
Full-pass transformations during training, but instance-level transformations during prediction.
Historical aggregations during training and prediction.
Historical aggregations during training, but real-time aggregations during prediction.

Preprocessing Operations

1) Instance-level transformations during training and prediction.

- These are straightforward transformations.
- Only values from the same instance are needed for the transformation.
- Examples:
 - clipping the value of a feature to some threshold,
 - polynomially expanding another feature,
 - multiplying two features,
 - or comparing two features to create a Boolean flag.
- These transformations must be applied identically during training and prediction.
 - Since the model will be trained on the transformed features, not on the raw input values.

Preprocessing Operations

2) Full-pass transformations during training, but instance-level transformations during prediction.

- These transformations are **stateful**; they use some precomputed statistics to perform the transformation.
- During training, you analyze the whole body of training data to compute quantities such as **minimum**, **maximum**, **mean**, and **variance** for transforming **training data**, **evaluation data**, and **new data** at prediction time.

Preprocessing Operations

2) Full-pass transformations during training, but instance-level transformations during prediction.

- For example, to normalize a numeric feature for training, you compute its **mean** (μ) and its **standard deviation** (σ) across the whole of the training data.
 - This computation is called a **full-pass** (or **analyze**) operation.
- At prediction time, the new instance is also normalized but using the same parameters estimated from the training dataset.
 - Don't compute new statistics from the test set to avoid *leaking information*.
 - Doing so affects the reliability of the test and evaluation results.

Preprocessing Operations

2) Full-pass transformations during training, but instance-level transformations during prediction.

- Full-pass transformations include the following

MinMax scaling numerical features using min and max computed from the training dataset.

Standard scaling (z-score normalization) numerical features using μ and σ computed on the training dataset.

Bucketizing numerical features using quantiles.

Imputing missing values using the median (numerical features) or the mode (categorical features).

Converting strings (nominal values) to integers (indexes).

Counting the occurrence of a term (feature value) in all the documents.

Computing the PCA of the input features to project the data into a lower dimensional space.

Preprocessing Operations

3) Historical aggregations during training and prediction.

- This involves creating business aggregations, derivations, and flags as input signals to the prediction task.
- These types of features can be precomputed and stored in a feature store to be used during model training.
- Example – predicting the customer value to promote a product for marketing. We need to compute and store the following values:
 - Recency – How recently did the customer purchase?
 - Frequency – How often do they purchase?
 - Monetary Value – How much do they spend?

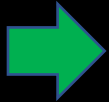
Preprocessing Operations

4) Historical aggregations during training, but real-time aggregations during prediction

- This involves creating a feature by summarizing real-time values over time.
- Example – building a model that **estimates the taxi trip time** based on the traffic metrics for the route in the last 5 minutes, in the last 10 minutes, in the last 30 minutes, and at other intervals.
- When the model for real-time prediction is served, the model expects features derived from the aggregated values as an input.
 - Therefore, you can use a stream-processing technology like **Apache Beam** to compute the aggregations from the real-time data points streamed into your system.

Content

Content
Introduction
Preprocessing Operations
Data Preprocessing techniques



Data Preprocessing Techniques

Why is Data Preprocessing Important?

Data Preprocessing Techniques

Why is Data Preprocessing Important?

- Because the algorithms learn from the data and the learning outcome depends on the proper data needed to solve a particular problem.
 - Most of the models can't handle missing values.
 - Some of them are affected by outliers, high dimensionality and noisy data.
- Always remember **garbage in, garbage out**.



Data Preprocessing Techniques

Important data preprocessing techniques

- Data Cleaning
- Dimensionality Reduction
- Feature Engineering
- Sampling Data
- Data Transformation
- Imbalanced Data

Data Preprocessing Techniques

Important data preprocessing techniques

- Data Cleaning
- Dimensionality Reduction
- Feature Engineering
- Sampling Data
- Data Transformation
- Imbalanced Data

Data Cleaning

- This technique refers to identifying incomplete, inaccurate, duplicated, irrelevant or null values in the data.
- After identifying these issues, you will need to either modify or delete them.
- There are three cases:
 - Noisy data
 - Missing data
 - Structural errors

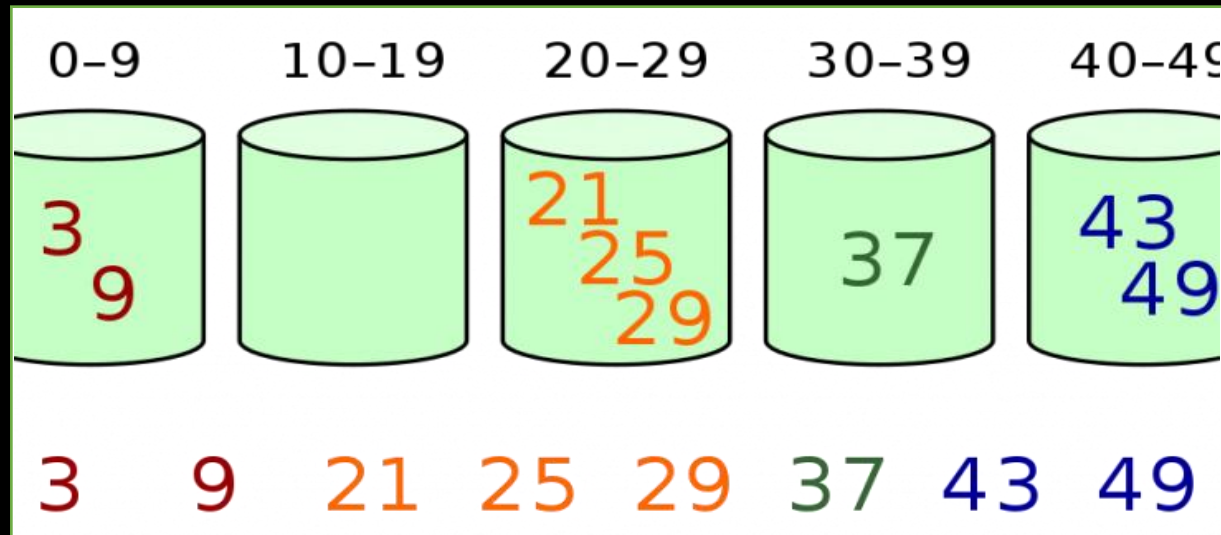
Data Cleaning

- **Noisy data** refers to meaningless data in your dataset.
 - incorrect records,
 - duplicated observations,
 - Outliers.
- Example: an “age” column has negative values.
- Example: say you need to predict whether a woman is pregnant or not. You don’t need the information about their hair color.

Data Cleaning

Solution:

- Binning (or bucketing):
 - First sort the values,
 - then divide them into “bins” (buckets with the same size),
 - and then apply a mean/median in each bin



Data Cleaning

Missing data

- Most machine learning models can't handle missing values in the data.
 - So, you need to intervene and adjust the data to be properly used inside the model.
- There are four solutions:

Removing the sample

- this is only recommended if:
 - You have a large dataset and a few missing records, so removing them won't impact the distribution of your dataset.
- Most of the attributes of that observation are null, so the observation itself is meaningless.

Fill using a global constant

- Fill using NAN or 0
- Fill using median
- Fill using mean
- Fill using mode

Backward/forward fill

- take the previous or next value to fill the missing value.

Filling using an ML algorithm

- Using KNN, first find the k instances closer to the missing value instance, and then get the mean of that attribute related to the k-nearest neighbors (KNN).
- Using regression, for each missing attribute, learn a regressor that can predict this missing value based on the other attributes.

Data Cleaning

Structural Errors

- Structural errors usually refer to some typos and inconsistencies in the values of the data.
- Example: an attribute describes the brand of a shoe; this attribute can be written in different ways:
 - Nike, NIKE, nike, nik, nikee, ...
- Solution: you can transform all words to lowercase letters, or removing additional letters and spaces.

Data Preprocessing Techniques

Important data preprocessing techniques

- Data Cleaning
- Dimensionality Reduction
- Feature Engineering
- Sampling Data
- Data Transformation
- Imbalanced Data

Dimensionality Reduction

- The dimensionality reduction is concerned with reducing the number of input features in training data. **(The Curse of Dimensionality)**
- Reducing the number of features can have a positive impact:
 - Requiring less computational resources
 - Increasing the overall performance of the model
 - Preventing overfitting
- Dimensionality reduction method:
 - Feature selection
 - Linear methods
 - Non-linear methods

Dimensionality Reduction

- Feature selection is the process of selecting the most important features related to the target value.
- Some techniques:
 - **Correlation Between Features:** drop features that have a high correlation with others.
 - **Statistical Tests:** checks the relationship of each feature individually with the output variable.
 - **Recursive Feature Elimination (RFE):** trains the model with all features in the dataset, calculating the performance of the model, and then drops one feature at a time.
 - **Variance Threshold:** detects features with high variability within the column, selecting those that got over the threshold.

Dimensionality Reduction

- **Linear methods** use linear transformations to reduce the dimensionality of the data.
- Techniques:
 - Principal Component Analysis (PCA)
 - Linear Discriminant Analysis (LDA)
 - Factor Analysis
- **Non-Linear methods** are used when the data doesn't fit in a linear space.
- Techniques:
 - The Multi-Dimensional Scaling (MDS)
 - The Isometric Feature Mapping (Isomap)

Data Preprocessing Techniques

Important data preprocessing techniques

- Data Cleaning
- Dimensionality Reduction
- Feature Engineering
- Sampling Data
- Data Transformation
- Imbalanced Data

Feature Engineering

- We use domain knowledge to create new features, which we manually generate from the existing features by applying some transformation to them.
- Examples:
 - **Decompose Categorical Attributes:** if a feature in your data about hair color and the values are “brown”, “blonde” and “unknown”. In this case, you can create a new column called “has color” and assign 1 if you get a color and 0 if the value is unknown.
 - **Decompose a DateTime:** Convert a date time column to more meaningful features, for example, period of the day (night or morning), day of the week, and so on.
 - **Reframe Numerical Quantities:** an example is applying mathematical operations between features.

Data Preprocessing Techniques

Important data preprocessing techniques

- Data Cleaning
- Dimensionality Reduction
- Feature Engineering
- Sampling Data
- Data Transformation
- Imbalanced Data

Sampling Data

- The more data you have, the greater the model's accuracy tends to be.
- Some machine learning algorithms can have difficulty handling a large amount of data and run into issues.
 - memory saturation, computational increase to adjust the model parameters, and so on.

Sampling Data

Sampling data techniques:

- **Sampling without replacement.** Selecting a data sample only once.
- **Sampling with replacement.** The data sample can be used multiple times.
- **Stratified sampling.** Splitting the data into many partitions and keep the proportional number of classes according to the original data.
- **Progressive sampling.** Starts with a small size and keeps increasing the dataset until a sufficient sample size is acquired.

Data Preprocessing Techniques

Important data preprocessing techniques

- Data Cleaning
- Dimensionality Reduction
- Feature Engineering
- Sampling Data
- Data Transformation
- Imbalanced Data

Data Transformation

- **Data transformations** is converting the data from one format to another.
- Some algorithms only works with numerical data, others work with categorical data, others can handle both types of data.
- Data transformation techniques:
 - **Transformation for Categorical Variables:** encoding categories into numerical representation
 - **Min-Max Scaler (Normalization):** scaling the data between a predefined range (usually between 0 and 1).
 - **Standardization (z-score normalization):** It transforms the data so that the mean of the data is zero and the standard deviation is one.

Data Preprocessing Techniques

Important data preprocessing techniques

- Data Cleaning
- Dimensionality Reduction
- Feature Engineering
- Sampling Data
- Data Transformation
- Imbalanced Data

Imbalanced Data

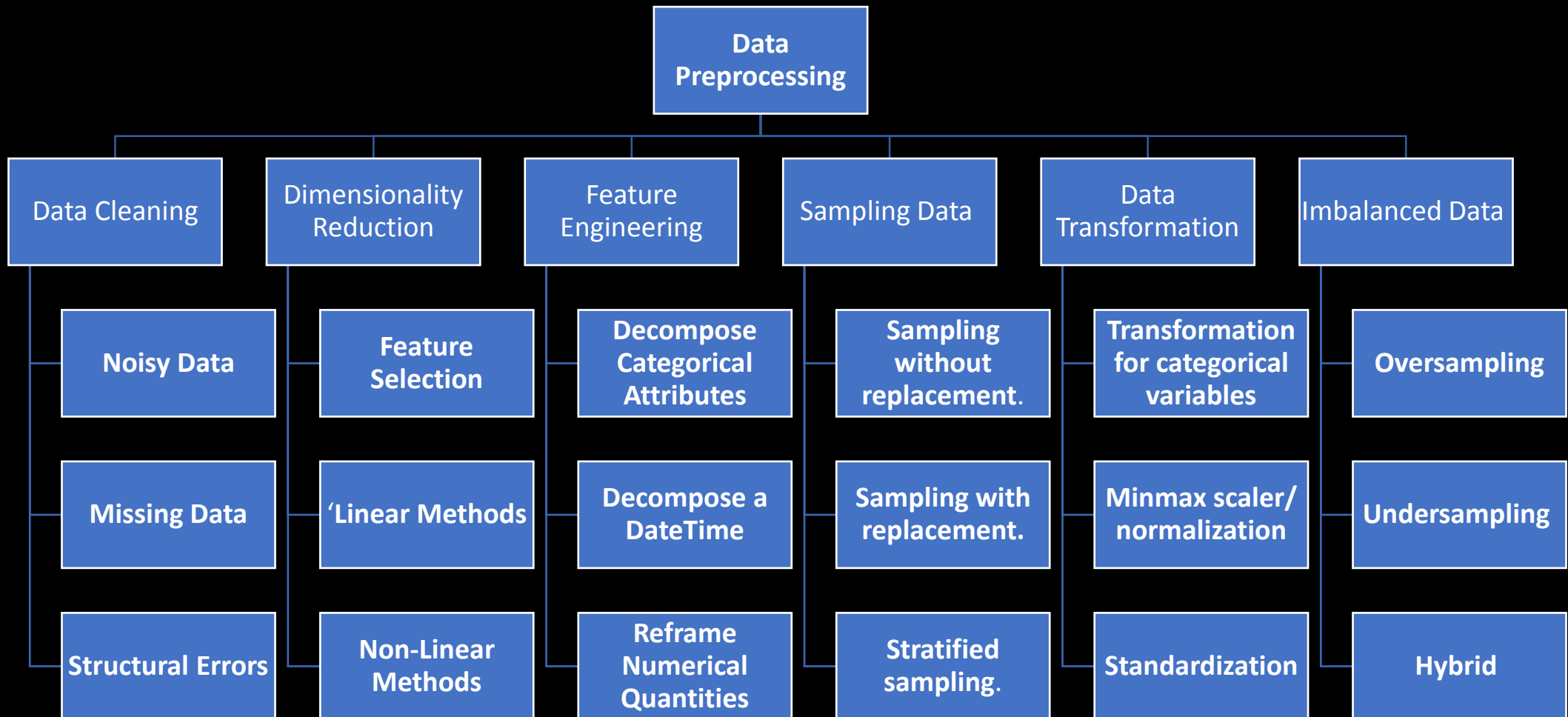
- Some datasets used for classification problems are **imbalanced**.
- Imbalanced data means that one of the classes has many examples that other classes.
- For example, for a dataset of spam/not-spam emails. If the spam emails are 150 samples and not-spam emails are 20 samples, then this data is “imbalanced”
- Iris dataset is a balanced dataset because each class (Setosa, Versicolor, Virginca) has 50 samples.

Imbalanced Data

Techniques for fixing imbalanced dataset:

- **Oversampling:** the process of increasing your dataset with synthetic data of the minority class.
- **Undersampling:** the process of reducing your dataset and removing real data from your majority class.
- **Hybrid:** combines the oversampling and undersampling techniques in your dataset.

Summary



References

- <https://cloud.google.com/architecture/data-preprocessing-for-ml-with-tf-transform-pt1>
- <https://www.scalablepath.com/data-science/data-preprocessing-phase>
- <https://medium.com/almabetter/data-preprocessing-techniques-6ea145684812>
- <https://developers.google.com/machine-learning/testing-debugging/pipeline/overview>