Feature Engineering

> **Feature engineering** is the process of transforming raw data into **features that better represent the underlying problem** to the predictive models, resulting in improved model accuracy on unseen data.

Feature engineering comprises several techniques, such as:

- **Feature selection - selecting** a subset of relevant features;
  - Reduce the data dimensionality;
  - Identifying and discarding **irrelevant and redundant** features;
- **Feature extraction - transform** existing variables into new ones;
  - Reduce the data dimensionality, but the dimensional space has a different nature, from the original one - **PCA (Principal Component Analysis)**;
- **Feature generation - create** new features from existing ones;
  - Creation of new variables, from subsets of the original ones.

Feature Selection

> Set of techniques with the goal of **reducing data dimensionality** by identifying the **necessary and sufficient variables** to describe the data and **discarding the irrelevant and redundant ones**.

- Choose δ variables from the original set of d variables, such that $\delta \leq d$;
- Particularly useful when the data is **sparse - many variables** and **few records**.

There are two types of feature selection techniques:

- The **unsupervised**, which is always possible to apply, and has the goal of **removing redundant variables**;

    - Two variables are **redundant** if they **convey the same information** about the target variable - **high correlation between them**;

- The **supervised**, which is only possible to apply when the target variable is available, and has the goal of **removing irrelevant variables**;
    - A variable is **irrelevant** if it has **no influence** on the target variable - **low correlation with the target variable**;

Feature selection techniques work as follow:

1. Receive a set of d variables, excluding the target variable (when available);
2. **Generate subsets** of variables, with δ variables, such that $\delta \leq d$, and **evaluate** the **quality** of each subset;
    (a) **Save** subsets that satisfy the **evaluation criteria**;
    (b) Discard the remaining subsets;
3. Go to step 2, until the **termination condition** is met.

Termination Condition

The termination condition is usually one of the following:

- Finding the **best number of variables to keep** - either a fixed number or a percentile;
- Specifying a **minimum quality threshold** - the evaluation criteria must be above a certain value - **ranking**.

Search Strategy

- The search strategy is responsible for **choosing which subsets to evaluate**;
- The **search space** is the set of all possible subsets of variables;
    - Since the search space is **exponential**, it is **impossible** to **evaluate all subsets - heuristics** are used to **reduce the search space**;
- The **Sequential Forward Selection (SFS)** is one of the most common search strategies:
    1. Start with an **empty subset**;
    2. **Add** the **best variable** to the subset;
    3. **Repeat** step 2, until the **termination condition** is met.
- **Sequential Backward Selection (SBS)** is the opposite of SFS:

1. Start with a **full subset**;
2. **Remove** the **worst variable** from the subset;
3. **Repeat** step 2, until the **termination condition** is met.

- Other algorithms exist, such as **genetic algorithms**.

To identify the **best subset of features** to train a classifier (supervised), we have two approaches:

- **Wrappers - evaluate** the **quality** of the **features together with the classifier**;
- **Filters - evaluate** the **quality** of the **features independently** of the **classifier**;
  - More **efficient** than wrappers.

Feature Extraction

> Apply **transformations to the original variables**, generation new ones, generally orthogonal to the original ones, with the goal of **reducing data dimensionality**.

There are several techniques for feature extraction:

- **Principal Component Analysis (PCA)**;
- **Singular Value Decomposition (SVD)**;
- **Linear Discriminant Analysis (LDA)**.

Principal Component Analysis (PCA)

- Find the set of variables that **best summarize the original data**, among all possible **linear combinations** of the original variables;
- Generates new variables as **linear combinations** of the original ones, creating a new space where the variables are **indepedent** and **orthogonal**;
- The importance of each variable is measured by the **variance** of the data in the direction of the variable;
- Works better after **scaling the data**;

The algorithm is as follows:

1. **Center the dataset**: $X = X - \mu^T$ - subtract the mean of each variable from the dataset;
2. **Compute the covariance matrix**: $\Sigma = \frac{1}{n-1}X^TX$ - the covariance matrix is a $d \times d$ matrix, where d is the number of variables;
3. **Compute the eigenvectors and eigenvalues of the covariance matrix**: $\Sigma v = \lambda v$ - the eigenvectors are the directions of the new space, and the eigenvalues are the variance of the data in the direction of the eigenvectors.

There are two ways to **choose the number of variables to keep**:

- Plot the explained variance ratio of each variable by descending order, and using the **elbow method** to choose the number of variables to keep;
- Choose the number of variables that explain a **minimum percentage of the variance - 90%** is a common value.

Singular Value Decomposition (SVD)

- Similar to PCA, but **works with non-square matrices**;
- Extracts more stable variables;
- Consists of **factorizing** a matrix D into three others:
  - L - left singular vectors;
  - δ - singular values;
  - R - right singular vectors;
- The **singular values** are the **square roots of the eigenvalues** of D^TD;
- The **left singular vectors** are the **eigenvectors** of DD^T;
- The **right singular vectors** are the **eigenvectors** of D^TD;
- The number of variables to use depends on the reconstruction error we are willing to accept - in general, we keep the number of variables to have a **reconstruction error of less than 10%**.

Linear Discriminant Analysis (LDA)

- Similar to PCA, but **works with labeled data - supervised**;
- **Maximizes the separation** between classes, while **minimizing the variance** inside each class;
- **The higher the variance within a class, the higher the probability of overlapping with other classes**;

- Drawbacks:
  - Data must be labeled, numeric and **normally distributed**;
  - The dimensionality must be **lower** than the number of records;
  - It is sensitive to outliers.

  Better classification results may be obtained if we **first apply PCA** to reduce data dimensionality, and **then apply LDA**, to maximize the separation between classes.

Feature Generation

> Create new variables from existing ones, derived from the **domain knowledge** of the problem.

Application of **any operation to any subset of variables**, either choosing them through **domain knowledge** or **automatically**.

- Usually used with feature selection;
- Alone, it always **increases data dimensionality**, but it can be used to **reduce data dimensionality** when combined with feature selection;
- Usually not used with feature extraction;
- Used to express **domain knowledge** in the data.

  A common approach is arithmetic operations between numeric variables.

Feature templates

Feature templates are the most usual mechanisms to create new variables; each one defines an operator specifying:

- The **variables** to use - **domain knowledge**;
- The **operation** to apply (sums, aggregations, splits, etc.);
- The **name** of the new variable;

Feature Stores

Create **repositories of variables**, from which the data scientist can choose the variables to use, instead of creating them from scratch. These stores have the following advantages:

- **Reusability** - variables can be reused in different projects;
- Variables generated only **once - less computation**.