**Feature engineering: the art of extracting the most from your data**

[[Automata](https://medium.com/@autognosi?source=post_page-----d9b59e395c59--------------------------------)](https://medium.com/@autognosi?source=post_page-----d9b59e395c59--------------------------------)

[[𝐀𝐈 𝐦𝐨𝐧𝐤𝐬.𝐢𝐨](https://medium.com/aimonks?source=post_page-----d9b59e395c59--------------------------------)](https://medium.com/aimonks?source=post_page-----d9b59e395c59--------------------------------)

[Automata](https://medium.com/@autognosi?source=post_page-----d9b59e395c59--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2Fa543da0e3424&operation=register&redirect=https%3A%2F%2Fmedium.com%2Faimonks%2Ffeature-engineering-the-art-of-extracting-the-most-from-your-data-d9b59e395c59&user=Automata&userId=a543da0e3424&source=post_page-a543da0e3424----d9b59e395c59---------------------post_header-----------)

Published in

[𝐀𝐈 𝐦𝐨𝐧𝐤𝐬.𝐢𝐨](https://medium.com/aimonks?source=post_page-----d9b59e395c59--------------------------------)

·

10 min read

·

Sep 11, 2023

99



Feature engineering by Author with ideogram.ai

**What is feature engineering and why is it essential for data science**

Feature engineering is the process of creating new variables from raw data that can improve the performance and interpretability of data science models. Variables, or features, are the characteristics of the data that we use to make predictions, classifications, clustering or other operations. Feature engineering involves transforming raw data into new features that are more informative and relevant to the problem being solved.

The benefits of feature engineering are many. First, feature engineering can improve model performance by increasing the accuracy and robustness of predictions. Additionally, feature engineering can reduce data complexity by eliminating irrelevant or redundant variables and simplifying data management. Finally, feature engineering can increase the interpretability of results by making the relationships between variables and their implications clearer.

Feature engineering skills are diverse and require both technical and creative skills. To do good feature engineering you need to have a good understanding of the data and the problem to be solved, know the application domain, the sources of the data and the objectives to be achieved. We must also use creativity and critical thinking to generate new ideas, hypotheses and solutions. Finally, the new variables must be tested and evaluated to verify their validity, quality and usefulness.

The difficulties of feature engineering are also many and depend on the context and situation. Feature engineering requires experience and intuition, there is no universal recipe or magic formula for creating the best variables. Feature engineering varies depending on the type and source of the data, which can be numeric, categorical, textual, temporal, or other. Feature engineering can also generate noise or redundancy, introducing errors or biases into data or models.

As you can see, feature engineering is a complex but crucial process for data science. Read on to find out what the main feature engineering techniques are and how to implement them.

**The main feature engineering techniques and how to implement them**

Feature engineering is based on several techniques that can be used to transform raw data into new variables. These techniques can be grouped into four main categories: variable selection, coding, generation and aggregation. Let’s see in detail what they are and how to implement them.

Variable selection consists of choosing the most relevant variables and reducing the dimensionality of the data. This technique is useful for eliminating irrelevant, redundant, or noisy variables that can degrade model performance or make data analysis more difficult. To select variables, different methods can be used, such as filtering, encapsulation or embeddedness. The filter method is based on statistical or informational criteria to evaluate the relevance of variables. The encapsulation method relies on a machine learning algorithm to select variables that optimize an objective function. The embedded method is based on a model that incorporates the selection of variables into the learning process.

Variable coding involves transforming existing variables to make them more compatible with models. This technique is useful for adapting variables to the type and shape required by models or for making variables more homogeneous and comparable. Different methods can be used to encode variables, such as scaling, normalization, standardization or binarization for numeric data, or one-hot encoding, target encoding, ordinal encoding or eli5 encoding for data categorical.

Variable generation involves creating new variables from existing ones or other information. This technique is useful for enriching data with new features that can better capture information relevant to the problem to be solved. To generate variables, different methods can be used, such as derivation, extraction, polynomization or interaction generation for numerical data, or bag-of-words, TF-IDF, word embedding or n-gram for textual data.

Variable aggregation is combining or condensing variables to create new properties. This technique is useful for creating new relationships between variables or for synthesizing information more effectively. To aggregate variables, different methods can be used, such as union, grouping, synthesis or generation of cross features for numerical data, or integration, fusion, concatenation or generation of n- gram for textual data.

As you can see, there are many feature engineering techniques you can use to transform your data. Read on to discover some practical examples applied to different types of data.

**Practical examples of feature engineering applied to different types of data**

Feature engineering can be applied to different types of data, depending on their characteristics and available techniques. In this section we will see some practical examples of feature engineering applied to four common data types: numeric, categorical, textual and temporal.

**Numeric data**

Numeric data are those that represent quantities, measurements, values or magnitudes. This data can be continuous or discrete, positive or negative, integer or decimal. Some examples of numeric data are age, income, weight, temperature, score, price, and so on.

To transform numerical data we can use different feature engineering techniques, such as:

* Scaling: consists of modifying the range of numerical data values to make them more homogeneous and comparable. For example, we can use min-max scaling to bring values between 0 and 1, or standardization to bring values to a mean of 0 and a standard deviation of 1.
* Discretization: consists of dividing the values of numerical data into discrete intervals or categories. For example, we can use discretization to transform age into age groups (young, adult, elderly) or income into income classes (low, medium, high).
* Polynomization: consists of creating new variables from numerical data by raising them to a power or combining them with each other. For example, we can use polynomization to transform a variable x into x², x³, or x\*y.
* The generation of interactions: consists of creating new variables from numerical data by multiplying them by another variable or by a coefficient. For example, we can use interaction generation to transform a variable x into x\_z or x\_0.5.

These techniques can help us make numerical data better fit models or better capture nonlinear relationships between variables.

**Categorical data**

Categorical data is that which represents qualities, attributes, classes, or groups. This data can be nominal or ordinal, binary or multiple, balanced or unbalanced. Some examples of categorical data are sex, nationality, color, vote, gender, and so on.

To transform categorical data we can use different feature engineering techniques, such as:

* One-hot coding: consists of transforming a categorical variable with n possible values into n binary variables that indicate the presence or absence of each value. For example, we can use one-hot encoding to transform the color to red (1 or 0), green (1 or 0), and blue (1 or 0).
* Target coding: consists of transforming a categorical variable with n possible values into a numerical variable that indicates the average of the target value for each value. For example, we can use target coding to turn genre into a numeric variable that indicates the average score for each genre.
* Ordinal coding: consists of transforming a categorical variable with n possible values into a numerical variable that indicates the order of the values. For example, we can use ordinal coding to transform the rating into a numeric variable indicating the approval level (1 to 5).
* The eli5 coding: consists of transforming a categorical variable with n possible values into a numerical variable that indicates the weight of each value with respect to the target value. For example, we can use eli5 coding to transform nationality into a numeric variable that indicates the importance of each nationality for predicting income.

**Textual data**

Textual data is data that represents words, sentences, texts or documents. This data can be structured or unstructured, short or long, simple or complex. Some examples of textual data are titles, reviews, emails, tweets, books, and so on.

To transform textual data we can use different feature engineering techniques, such as:

* The bag-of-words: consists of transforming a text into a vector of frequencies of the words that compose it. For example, we can use the bag-of-words to transform the sentence “I like ice cream” into a vector indicating how many times the words “I”, “like” and “ice cream” appear.
* The TF-IDF: consists of transforming a text into a vector of weights of the words that compose it, based on their frequency in the text and in the collection of documents. For example, we can use TF-IDF to transform the sentence “I like ice cream” into a vector that indicates how important the words “I,” “like,” and “ice cream” are to the meaning of the sentence.
* Word embedding: consists of transforming a word into a vector of real numbers that capture its semantic and syntactic meaning. For example, we can use word embedding to transform the word “ice cream” into a vector representing its flavor, temperature, shape and associations.
* The n-gram: consists of transforming a text into a sequence of n consecutive words that capture its structure and context. For example, we can use the n-gram to transform the sentence “I like ice cream” into a sequence of bi-grams (“I like”, “like the”, “the ice cream”) or tri-grams (“I like il”, “like ice cream”).

**Temporal data**

Temporal data are those that represent events, measurements, values or quantities that vary over time. This data can be continuous or discrete, regular or irregular, stationary or non-stationary. Some examples of temporal data are time series, time series, sensor data, traffic data, and so on.

To transform temporal data we can use different feature engineering techniques, such as:

* Decomposition: consists of breaking down a time series into three components: trend, seasonality and residual. For example, we can use decomposition to decompose a time series representing monthly sales of a product into a component indicating the general trend of sales, a component indicating seasonal variation in sales, and a component indicating random fluctuations in sales .
* Differentiation: consists of transforming a time series into a series of differences between consecutive values or between values separated by an interval. For example, we can use differencing to transform a time series representing the daily price of a stock into a series of differences between daily prices or between weekly prices.
* Seasonality: consists of transforming a time series into a series of binary indicators that indicate the presence or absence of a seasonal effect. For example, we can use seasonality to transform a time series representing daily energy consumption into a series of binary indicators that indicate whether the day is a holiday or a work day.
* Harmonicity: consists of transforming a time series into a series of coefficients that indicate the presence and intensity of periodic cycles. For example, we can use harmonicity to transform a time series representing the average daily temperature into a series of coefficients indicating the presence and intensity of annual or daily cycles.

**Tips and resources to learn more about feature engineering and become an expert**

Feature engineering is a skill that can be learned, applied, improved and shared. In this section I will give you some tips and resources to learn more about the topic and become an expert in feature engineering.

**How to learn feature engineering**

To learn feature engineering you need to know the theoretical principles, best practices, case studies and challenges to face. I recommend you follow these steps:

* Study the fundamental concepts of feature engineering, such as definition, benefits, requirements, and challenges.
* Practice the main techniques of feature engineering, such as variable selection, coding, generation and aggregation.
* Explore real-world case studies of feature engineering, such as those applied to different types of data or different data science problems.
* Address feature engineering challenges, such as those related to data quality, complexity, creativity, or evaluation.

**How to apply feature engineering**

To apply feature engineering you need to follow steps, use tools, monitor metrics and make evaluations. I recommend you follow these steps:

* Define the problem to solve and the objectives to achieve with feature engineering. For example, you can define whether you want to make a prediction, classification, clustering or other, and what the target variables and predictor variables are.
* Explore the available data and analyze their characteristics, qualities and relationships. For example, you can explore data type, shape, distribution, correlation, and missingness.
* Choose the feature engineering techniques best suited to your problem and your data. For example, you can choose whether to use variable selection, coding, generation, or aggregation, or a combination of these.
* Implement feature engineering techniques using the most appropriate tools for your case. For example, you can use programming languages like Python or R, or specific libraries like scikit-learn or pandas.
* Monitor the performance metrics of your models and evaluate the impact of feature engineering on results. For example, you can monitor the accuracy, precision, recall, or F1 value of your models, and evaluate whether feature engineering has improved or degraded performance.

**How to improve feature engineering**

To improve feature engineering you need to know advanced techniques, the latest news, sources of inspiration and learning opportunities. I recommend you follow these steps:

* Study advanced feature engineering techniques, such as those based on machine learning, optimization or statistics. For example, you can study how to use automated feature selection, neural network feature extraction, Bayesian feature engineering, or causal feature engineering.
* Follow the latest news and trends in feature engineering, such as new methods, tools, applications or challenges. For example, you can follow blogs, podcasts, webinars, or newsletters dedicated to feature engineering
* Find sources of inspiration and creativity for feature engineering, such as those from other domains, problems, or data. For example, you can find ideas for feature engineering by watching videos, reading books, attending workshops, or visiting museums.
* Seize learning and improvement opportunities for feature engineering, such as those offered by online courses, books, certifications or mentors. For example, you can deepen your knowledge and skills about feature engineering by taking online courses

**Conclusion**

In this article I showed you what feature engineering is, why it is fundamental for data science, what are the main techniques to use and how to apply them to different types of data. I’ve also given you some tips and resources to delve deeper into the topic and become an expert in feature engineering.

Feature engineering is one of the most important and creative skills for a data scientist. It consists of creating new variables from raw data that can improve the performance and interpretability of the models. Feature engineering requires a good understanding of the data and the problem to be solved, the use of creativity and critical thinking, and the testing and evaluation of new variables. Feature engineering is based on different techniques, such as variable selection, coding, generation and aggregation, which can be applied to different types of data, such as numeric, categorical, textual and temporal data.

I hope this article was useful and interested you. If you want to learn more about feature engineering, I recommend you follow the links I have included in the text, where you will find excellent resources to learn, apply, improve and share your feature engineering. Thanks for reading my article and happy feature engineering!