# feature transformation

## Author and editors

Alexander Jung*, Department of Computer Science, Aalto University, Espoo, Finland (ORCID 0000-0001-7538-0990)
Editors: Konstantina Olioumtsevits (ORCID 0000-0003-2697-8659), Ekkehard Schnoor (ORCID 0000-0001-8252-1499), Aalto University, Espoo, Finland
*Corresponding author: alex.jung@aalto.fi

## Synonyms

feature map, basis expansion

## Abstract

A feature transformation is a function that maps the feature vector of a data point to a new feature vector, so that a machine learning (ML) method runs on the new features in place of the raw ones. What is learned is then the composition of the transformation with a hypothesis on the new feature space, which is why a linear model of the new features is not a linear model of the raw ones. Methods differ in where the transformation comes from: principal component analysis (PCA) obtains it from the raw features alone, linear discriminant analysis (LDA) uses the labels as well, a random projection draws it at random, and an autoencoder learns it with an artificial neural network (ANN). Each hidden layer of a deep net is one more feature transformation, and a kernel method fixes one through the choice of a kernel without ever evaluating it. The number of new features sizes the hypothesis space of the method that runs on them, so keeping it small is model pruning, one of the three routes that regularization distinguishes. For the maximum daytime temperature at Krems an der Donau (Austria), the first five powers of the day of the year turn a nearly flat hypothesis into a curve that follows the seasons and cut the validation error from 88.2 to 23.6 squared degrees. The choice matters as much as the count: the three odd powers of the day leave the validation error above that of the constant hypothesis, which is why the transformation is often learned from the dataset rather than chosen by hand.

## Definition

A weather station at Krems an der Donau (Austria) records the maximum daytime temperature of every day of 2023 and 2024. Each day is a data point, its label is that temperature, and its single raw feature $x$ is the position of the day in the year, rescaled to the interval $[-1, 1]$. The 365 days of 2023 form the training set $\mathcal{D}^{(\text{train})}$ and the 366 days of 2024 the validation set $\mathcal{D}^{(\text{val})}$. Linear regression on $x$ learns a nearly flat hypothesis from $\mathcal{D}^{(\text{train})}$ by least squares: it rises by 6.9 degrees from the first day of the year to the last, against a label range

of 38.4 degrees. Its training error is 79.4 squared degrees and its validation error 88.2, above the 84.9 of the constant hypothesis that always predicts the sample mean of the labels in $\mathcal{D}^{(\text{train})}$. Replacing $x$ by the five numbers $x, x^2, \ldots, x^5$ and running the same method on them gives a curve that follows the seasons, with a validation error of 23.6. Fig. 1(a) draws the two learned hypotheses over the days of $\mathcal{D}^{(\text{train})}$.

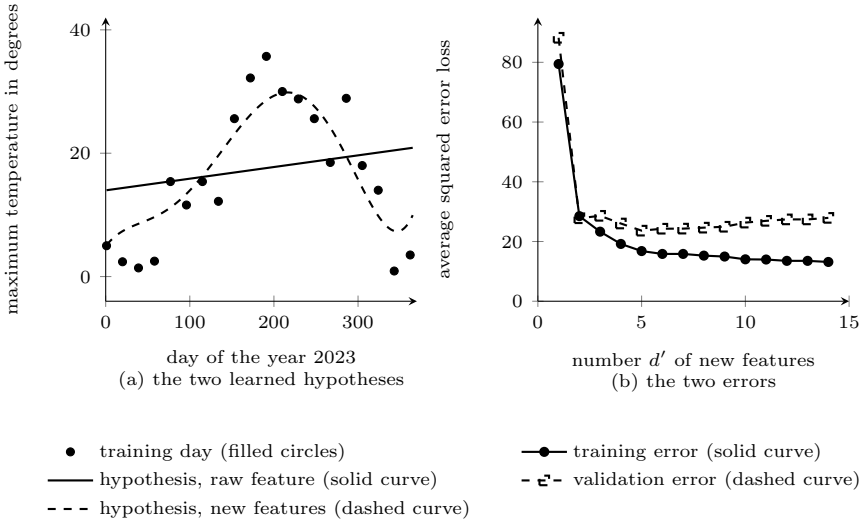



Figure 1: A feature transformation of the single raw feature of a day at Krems, its position in the year. (a) Twenty of the 365 days of $\mathcal{D}^{(\text{train})}$, every 19th one in calendar order (filled circles), the hypothesis learned from the raw feature alone (solid curve), and the hypothesis learned from the $d' = 5$ new features (dashed curve), which is a polynomial of the raw feature. (b) The training error (solid curve, filled circles) and the validation error (dashed curve, open squares) against the number $d'$ of new features. Data generated by `pythondemos/featuretransformation.py`

A feature transformation is a function

$$\mathbf{\Phi} : \mathcal{X} \to \mathcal{X}', \qquad \mathbf{x} \mapsto \mathbf{\Phi}(\mathbf{x}), \tag{1}$$

which carries the feature vector $\mathbf{x}$ of a data point from its feature space $\mathcal{X}$ to a new feature vector in a new feature space $\mathcal{X}'$. For a single raw feature, the powers of the opening example are the transformation

$$\mathbf{\Phi}(x) := \big(x, x^2, \ldots, x^{d'}\big)^\top, \tag{2}$$

which delivers $d'$ new features. A machine learning (ML) method then runs on the new features in place of the raw ones, and it delivers a hypothesis $h'$ on $\mathcal{X}'$. What is learned overall is then the composition

$$h = h' \circ \mathbf{\Phi} : \mathcal{X} \to \mathcal{Y}, \qquad \mathbf{x} \mapsto h'\big(\mathbf{\Phi}(\mathbf{x})\big). \tag{3}$$

The composition (3) is why a linear model of the new features is not a linear model of the raw ones. With the powers (2), the learned $h$ is a polynomial of degree $d'$ in $x$, which is polynomial regression (Bishop, 2006, Sect. 3.1).

Methods differ in where $\mathbf{\Phi}$ comes from. Principal component analysis (PCA) obtains it from the raw features alone, as the linear map of a given rank whose reconstruction error is smallest (Hastie et al., 2009, Sect. 14.5.1). Linear discriminant analysis (LDA) uses the labels as well, projecting onto the subspace spanned by the class centroids (Hastie et al., 2009, Sect. 4.3.3). A random projection draws $\mathbf{\Phi}$ at random instead of learning it (Johnson and Lindenstrauss, 1984), and an autoencoder learns a nonlinear $\mathbf{\Phi}$ with an artificial neural network (ANN) (Hinton and Salakhutdinov, 2006). Each hidden layer of a deep net is one more feature transformation, learned jointly with the output layer (Mallat, 2016). A kernel method fixes $\mathbf{\Phi}$ through the choice of a kernel, which returns the inner product of two transformed feature vectors (Schölkopf and Smola, 2002, Sect. 1.1). Such a method uses the transformed feature vectors only through those inner products, so $\mathbf{\Phi}$ is never evaluated (Schölkopf and Smola, 2002, Sect. 1.5). A transformation that delivers fewer new features than the data point has raw ones is dimensionality reduction, and obtaining $\mathbf{\Phi}$ from a dataset rather than by hand is the subject of feature learning.

The number $d'$ of new features in the definition (1) sizes the hypothesis space of the method that runs on them. For the powers (2), every hypothesis available at $d'$ is available at $d' + 1$ as well, so the training error, the average squared error loss over $\mathcal{D}^{(\text{train})}$, cannot rise with $d'$: it falls from 79.4 to 13.2 squared degrees as $d'$ grows from 1 to 14. The validation error on $\mathcal{D}^{(\text{val})}$ behaves differently. It is smallest at $d' = 5$ with 23.6 squared degrees and rises again to 27.9 at $d' = 14$, the overfitting that a large hypothesis space brings (Bishop, 2006, Sect. 1.1). Fig. 1(b) draws the two errors. Keeping $d'$ small therefore shrinks the hypothesis space, which is model pruning, one of the three routes that regularization distinguishes.

Two other names are in use for the function (1). The kernel method literature calls it a feature map (Schölkopf and Smola, 2002, Sect. 2.1), and the statistical learning literature a basis expansion (Hastie et al., 2009, Sect. 5.1). The two senses of *feature map* are distinct: a feature transformation is the map between feature spaces of the definition (1), while a feature map is the grid of activations delivered by one channel of a convolutional layer.

Since the training error cannot rise with $d'$, it cannot report the cost of a transformation that delivers too many new features, so $d'$ is a hyperparameter and is chosen on held-out data points by validation. The count is not the whole choice, since a transformation fixed before the labels are looked at can discard what the label depends on. The three odd powers $x, x^3, x^5$ leave the validation error at 86.7 squared degrees, above the 84.9 of the constant hypothesis, while the two even powers $x^2, x^4$ alone give 22.0. Every odd power is antisymmetric about $x = 0$, the middle of the year, and the maximum daytime temperature at Krems is nearly symmetric about that point. The composition (3) also changes what a weight means: a weight of a linear model on the new features multiplies one of them, at Krems a cube of a rescaled calendar position, and not a quantity

the station recorded.

In return, least squares on the new features reaches hypotheses that no linear model of the raw feature contains, at Krems a curve that follows the seasons where a straight line is of no use. Two alternatives avoid the choice of $\mathbf{\Phi}$ by hand. One keeps the raw features and takes a method whose hypothesis space is not a linear model of them, such as a decision tree. The other obtains $\mathbf{\Phi}$ from the dataset, which is feature learning, with an autoencoder or the hidden layers of a deep net.

## Cross References

feature learning, feature, feature space, PCA, LDA, random projection, autoencoder, dimensionality reduction, kernel method, feature map, polynomial regression, regularization, hyperparameter, validation, decision tree

## References


1. Bishop (2006). Pattern Recognition and Machine Learning. Springer Science+Business Media, New York, NY, USA.

2. Hinton and Salakhutdinov (2006). Reducing the dimensionality of data with neural networks. Science, 313(5786), pp. 504–507.

3. Schölkopf and Smola (2002). Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge, MA, USA.

4. Mallat (2016). Understanding deep convolutional networks. Philos. Trans. Roy. Soc. A, 374(2065), pp. Art. no. 20150203.

5. Hastie et al. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed., Springer Science+Business Media, New York, NY, USA.

6. Johnson and Lindenstrauss (1984). Extensions of Lipschitz mappings into a Hilbert space. Contemporary Math., 26, pp. 189–206.