What is dimensionality reduction?

Dimensionality reduction techniques such as PCA, LDA and t-SNE enhance machine learning models. They preserve essential features of complex data sets by reducing the number predictor variables for increased generalizability.

Dimensionality reduction is a method for representing a given dataset using a lower number of features (that is, dimensions) while still capturing the original data's meaningful properties.1 This amounts to removing irrelevant or redundant features, or simply noisy data, to create a model with a lower number of variables. Dimensionality reduction covers an array of feature selection and data compression methods used during preprocessing. While dimensionality reduction methods differ in operation, they all transform high-dimensional spaces into low-dimensional spaces through variable extraction or combination.

Why use dimensionality reduction?

In machine learning, dimensions (or features) are the predictor variables that determine a model's output. They may also be called input variables. High-dimensional data denotes any dataset with a large number of predictor variables. Such datasets can frequently appear in biostatistics, as well as social science observational studies, where the number of data points (that is, observations) outweighs the number of predictor variables.

High-dimensional datasets pose a number of practical concerns for machine learning algorithms, such as increased computation time, storage space for big data, and so on. But the biggest concern is perhaps decreased accuracy in predictive models. Statistical and machine learning models trained on high-dimensional datasets often generalize poorly.

Curse of dimensionality

The curse of dimensionality refers to the inverse relationship between increasing model dimensions and decreasing generalizability. As the number of model input variables increase, the model's space increases. If the number of data points remains the same, however, the data becomes sparse. This means the majority of the model's feature space is empty, that is, without observable data points. As data sparsity increases, data points become so dissimilar that predictive models become less effective at identifying explanatory patterns.2

In order to adequately explain patterns in sparse data, models may overfit on training data. In this way, increases in dimensionality can lead to poor generalizability. High-dimensionality can further inhibit model interpretability by inducing multicollinearity. As the quantity of model variables increase, so does the possibility that some variables are redundant or correlated. Collecting more data can reduce data sparsity and thereby offset the curse of dimensionality. As the number of dimensions in a model increase, however, the number of data points needed to impede the curse of dimensionality increases exponentially.3 Collecting sufficient data is, of course, not always feasible. Thus, the need for dimensionality reduction to improve data analysis.

Dimensionality reduction methods

Dimensionality reduction techniques generally reduce models to a lower-dimensional space by extracting or combining model features. Beyond this base similarity, however,

dimensionality reductions algorithms vary.

Principal component analysis

Principal component analysis (PCA) is perhaps the most common dimensionality reduction method. It is a form of feature extraction, which means it combines and transforms the dataset's original features to produce new features, called principal components. Essentially, PCA selects a subset of variables from a model that together comprise the majority or all of the variance present in the original set of variables. PCA then projects data onto a new space defined by this subset of variables.4

For example, imagine we have a dataset about snakes with five variables: body length (*X1*), body diameter at widest point (*X2*) fang length (*X3*), weight (*X4*), and age (*X5*). Of course, some of these five features may be correlated, such as body length, diameter and weight. This redundancy in features can lead to sparse data and overfitting, decreasing the variance (or generalizability) of a model generated from such data. PCA calculates a new variable (*PC1*) from this data that conflates two or more variables and maximizes data variance. By combining potentially redundant variables, PCA also creates a model with less variables than the initial model. Thus, since our dataset started with five variables (that is, five-dimensional), the reduced model can have anywhere from one to four variable (that is, one- to four-dimensional). The data is then mapped onto this new model.5

This new variable is none of the original five variables but a combined feature computed through a linear transformation of the original data's covariance matrix. Specifically, our combined principal component is the eigenvector corresponding to the largest eigenvalue in the covariance matrix. We can also create additional principal components combining other variables. The second principal component is the eigenvector of the second-largest eigenvalue, and so forth.6

Linear discriminant analysis

Linear discriminant analysis (LDA) is similar to PCA in that it projects data onto a new, lower dimensional space, the dimensions for which are derived from the initial model. LDA differs from PCA in its concern for retaining classification labels in the dataset. While PCA produces new component variables meant to maximize data variance, LDA produces component variables that also maximize class difference in the data.7

Steps for implementing LDA are similar to those for PCA. The chief exception is that the former uses the scatter matrix whereas the latter uses the covariance matrix. Otherwise, much as in PCA, LDA computers linear combinations of the data's original features that correspond to the largest eigenvalues from the scatter matrix. One goal of LDA is to maximize interclass difference while minimizing intraclass difference.8

T-distributed stochastic neighbor embedding

LDA and PCA are types of linear dimensionality reduction algorithms. T-distributed stochastic neighbor embedding (t-SNE), however, is a form of non-linear dimensionality reduction (or, manifold learning). In aiming to principally preserve model variance, LDA and PCA focus on retaining distance between dissimilar datapoints in their lower dimensional representations. In contrast, t-SNE aims to preserve the local data structure with reducing model dimensions. t-SNE further differs from LDA and PCA in that the latter two may produce models with more than three dimensions, so long as their generated model has less dimensions than the original data. t-SNE, however, visualizes all datasets in either three or two dimensions.

As a non-linear transformation methods, t-SNE foregoes data matrices. Instead, t-SNE utilizes a Gaussian kernel to calculate pairwise similarity of datapoints. Points near one another in the original dataset have a higher probability of being near one another than those further away. t-SNE then maps all of the datapoints onto a three or two-dimensional space while attempting to preserve data pairs.9

There are a number of additional dimensionality reduction methods, such as kernel PCA, factor analysis, random forests, and singular value decomposition (SVD). PCA, LDA, and t-SNE are among the most widely used and discussed. Note that several packages and libraries, such as scikit-learn, come preloaded with functions for implementing these techniques.

Example use cases

Dimensionality reduction has often been employed for the purpose of data visualization.

Biostatistics

Dimensionality reduction often arises in biological research where the quantity of genetic variables outweigh the number of observations. As such, a handful of studies compare different dimensionality reduction techniques, identifying t-SNE and kernel PCA among the most effective for different genomic datasets.10 Other studies propose more specific criterion for selecting dimensionality reduction methods in computational biological research.11 A recent study proposes a modified version of PCA for genetic analyses related to ancestry with recommendations for obtaining unbiased projections.12

Natural language processing

Latent semantic analysis (LSA) is a form of SVD applied to text documents natural language processing. LSA essentially operates on the principle that similarity between words manifests in the degree to which they co-occur in subspaces or small samples of the language.13 LSA is used to compare the language of emotional support provided by medical workers to argue for optimal end-of-life rhetorical practices.14 Other research uses LSA as an evaluation metric for confirming the insights and efficacy provided by other machine learning techniques.15