Műszaki Tudományos Közlemények vol. 13. (2020) 148–151.
DOI: Hungarian: https://doi.org/10.33895/mtk-2020.13.27
English: https://doi.org/10.33894/mtk-2020.13.27

DIMENSIONALITY REDUCTION METHODS USED IN MACHINE LEARNING

Kristóf MUHI,[1] Zsolt Csaba JOHANYÁK[2]

John von Neumann University, GAMF Faculty of Engineering and Computer Science, Kecskemét, Hungary

[1] *muhi.kristof@gamf.uni-neumann.hu*

[2] *johanyak.csaba@gamf.uni-neumann.hu*

Abstract

In most cases, a dataset obtained through observation, measurement, etc. cannot be directly used for the training of a machine learning based system due to the unavoidable existence of missing data, inconsistencies and high dimensional feature space. Additionally, the individual features can contain quite different data types and ranges. For this reason, a data preprocessing step is nearly always necessary before the data can be used. This paper gives a short review of the typical methods applicable in the preprocessing and dimensionality reduction of raw data.

Keywords*: machine learning, dimension reduction, data processing, big data.*

1. Data pre-processing

1.1. Data cleaning

When performing learning that is based on sample data, the available data set is called a multitude of samples. Each sample is described by a record or vector, that is, a set of values that contains values for different characteristics of that sample. For example, the *KDD Cup* 99 [1] database has 41 observed characteristics and 4,898,431 data records.

During data clean-up, we filter out records where the value of a property is not of the allowed type (eg., incorrect protocol identifier). Additionally, data cleansing can be used to remove duplicates, as duplicate data can result in erroneous, misleading statistics.

1.2. Missing data

Some records in the resulting data set may be incomplete (for example one or more attribute values were not recorded). In such cases, one of the following strategies should be followed.

1.2.1. Ignoring the affected records (vectors)

This is the simplest solution. If a few records are incomplete in a sample set of thousands (eg. the number of affected records is below 1%), one can usually leave out the affected records at no particular risk. Before deciding on this option, it is also worth considering whether there is no signal value for a given attribute value, for example in the case of network traffic logging the affected data could not be captured due to an attack

1.2.2. Replace missing data with the average value for that attribute

This is a simple solution that is relatively easy to implement, but in return there may be a high risk of completely erroneous inferences from the resulting dataset.

1.2.3. One by one to fill in missing data using other sources

The solution may be realistic for a few dozen affected records, but it is very costly and time consuming. A prerequisite for its implementation is that we have some a priori knowledge of the subject matter.

1.2.4. To fill in missing data using regression/ interpolation techniques

The method can be used when some regularity (for example linear change) is observed between the values of the given attribute observed in successive records or the actual attribute is a "dependent attribute", ie. its current value is derived from the values of another attribute or attributes. If one succeeds in identifying a dependent attribute, in most cases it is not worth putting too much energy into replacing the missing values of the attribute, as this attribute will be lost during dimension reduction.

1.3. Dependent attributes

An attribute is called dependent if its value is clearly derived from the value of one or more other attributes. The value of the dependent attribute is a redundant data, which only increases the amount of data to be handled and thus the time and memory required for the calculations. In dimension reduction, we also aim to filter out dependent attributes.

1.4. Convert to numeric form

We need numerical data to evaluate similarity between samples and calculate different statistics. If the order / spacing of the values in the labels / categories / different text data is clear (for example, even), the task can be easily solved. For example, suppose we have three possible label values. They are small, medium and large. The sequence is clear, and one can assume that the consecutive values are at the same distance. In this case, the occurrences of the three labels are replaced by values of 1, 2, and 3.

1.5. Cardinality reduction

A common problem with data model building is that we have to work with very large amounts of data. The availability of large amounts of data is useful, but it can increase the amount of time it takes to build a model, so in many cases we need to try to adjust the amount of data used to an optimum point while keeping as much information as possible from the original data set, i.e. the subset selected must be representative [2].

2. Dimension reduction

The data records available during model building often contain a large number of attributes, which is partly due to the fact that at the time of data collection planning, one did not have sufficient information on the observed phenomenon, thus they decided to gather all imaginable, recordable data to get a picture and not miss something important.

However, many features also require a large amount of memory and computing, which can make the task insurmountable. Therefore, we must strive to reduce the dimension of the task. The purpose of dimension reduction methods is to map the original m dimensional data points into a k dimensional space (where $k < m$), such that [3]

- preserving the divisibility of the points as much as possible, that is, the points belonging to one class / category should be more distinct from the points belonging to the other classes / categories;
- minimize information loss.

There are many ways to do this. First, let's look at those that do not entail a loss of information, that is, aiming at filtering out redundant data.

2.1. Decrease dimension without loss of information

For datasets that are created on the basis of the "we collect everything" principle, it is easy for a feature to have the same value in every record. In this case, this feature (or column, if you think of a table / matrix) is removed from our database without further ado. Although this seems a trivial solution, it can easily occur in practice (see the NLS-KDD [4] 20 % training database).

2.2. Dimension reduction by principal component analysis

For most practical applications, a small amount of information loss is an acceptable risk if, in return, a significant reduction in the number of dimensions can be achieved. Consider the sample data available to us as points in a multidimensional space. Then each element of the sample data record represents a coordinate of the point in this space. For many tasks, these points are not randomly spaced, but follow some regularity, or the points are not uniform in all directions.

For example, in Figure 1 the dots are located approximately along a straight line. By plotting this imaginary line as a coordinate axis (c_1 in Figure 1) and plotting the perpendicular coordinate axis, it can be observed that the points show a high degree of variance along the c1 axis and have relatively little perpendicular to it (c_2).

If we plot the orthogonal projections of the dots on the c1 axis, we obtain a score on the axis (red dots in Figure 2). These points differ only slight-

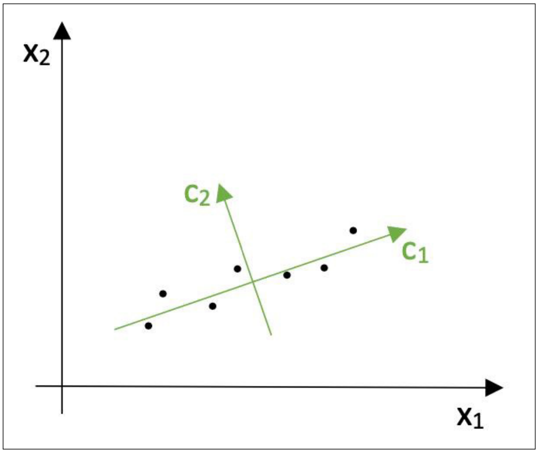

Figure 1. *Set of points with new coordinate axes.*

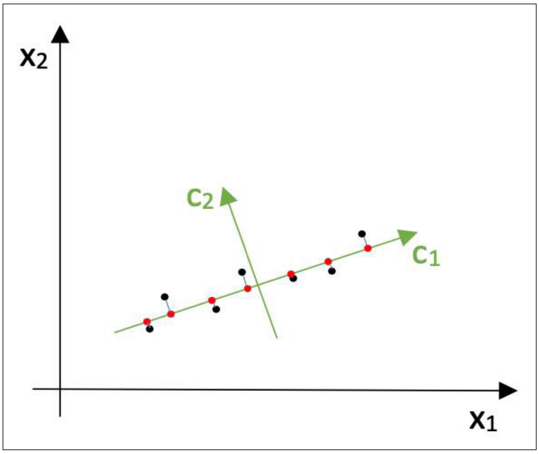

Figure 2. *Projection on the c_1 axis.*

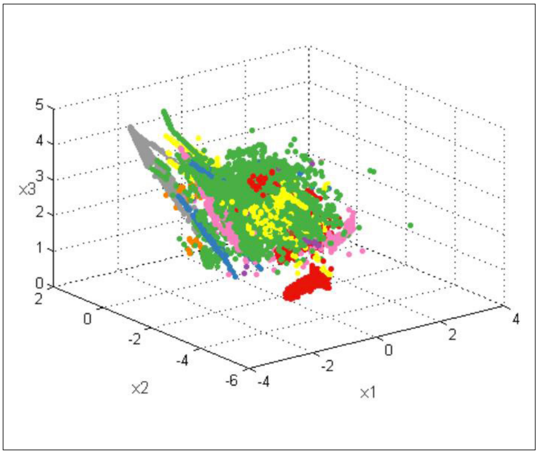

Figure 3. *Random projection of NSL-KDD 20% learning dataset into three-dimensional space*

ly from the original points, but their position can be described by a single coordinate along the c_1 axis. By working with these points instead of the originals, the dimension number of the originally two-dimensional data set is reduced by one with a small amount of data loss.

The principal component analysis aims to identify these axes and then determine the coordinates of the points in the new coordinate system by means of a linear transformation.

2.3. Information gain based dimension reduction

The basic idea of the Information Gain (IG) based method of dimension reduction is related to decision tree theory (eg, ID 3, C4.5, C5.0 algorithms [5]) where in the case of each iteration a dataset is divided into subsets according to the values of the selected characteristic. The algorithm is based on entropy calculation. The entropy of the set *S*

$$E(S) = -\sum_{k=1}^{N} p_k \, log_2 \, p_k \qquad (1)$$

characterizes the "inhomogeneity", that is, the variability of the set. Here N is the number of values in the set and p_k is the relative frequency of each value given by

$$p_k = \frac{n_k}{n} \qquad (2)$$

where *n* is the total set of elements, n_k is the number of sets having the label *k*.

2.4. Random Projection

This method projects the points from the originally m-dimensional space to a lower (r) dimensional -space using a random linear projection. The projection is done by multiplying the feature matrix by a matrix of random numbers.

$T = X \cdot RP$, where $T \in \mathbb{R}^{nxr}$, $X \in \mathbb{R}^{nxm}$, $RP \in \mathbb{R}^{mxr}$ and *RP* each row has unit size vector

$$\| RP_i \|^2 = 1, j = 1..m.$$

The random numbers follow the Gaussian distribution. The method preserves the distance between points well and requires less computation than PCA. The quality of the transformation depends on the number of points and the value of *r*. Figure 3 shows a 3D random projection of the NSL-KDD 20% training data set. The colours of the dots refer to traffic types.

2.5. t-Distributed Stochastic Neighbor Embedding)

The t-DSNE method tends to keep similar points close to one another while reducing dimensions by reducing the number of dimensions [3]. The original points are modeled as coming from a normal distribution and the nested points as if they were from a Student (t) distribution. The method is mostly used for two- or three-dimensional visualization of point groups (clusters). The disadvantage of this method is that it does not result in a matrix or formula that would allow other (e.g., test, validation, etc.) data to be transformed into the same space from the original m-dimensional space. **Figure 4.** shows the result of the t-DSNE transformation for 10,000 samples randomly selected from the NSL-KDD 20% learning data set.

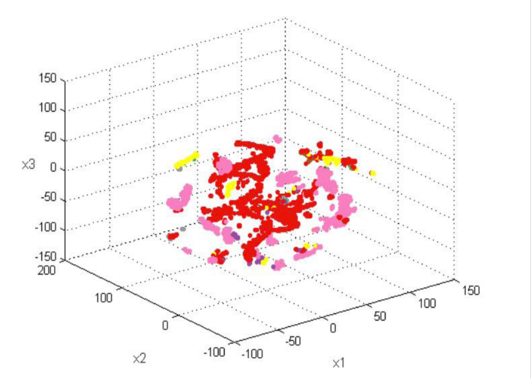

Figure 4. *Transformation of 10,000 samples t-DSNE randomly selected from the NSL-KDD 20% training data set into 3D space using Chebysev distances*

3. Conclusions

In most of the cases, raw sample data to be used in machine learning must undergo a pre-processing step prior to actual use. This step involves addressing the problem of missing data, reducing cardinality and dimensionality. In this article we have briefly reviewed the solutions most commonly used in such cases.

Acknowledgment

This research is supported by EFOP-3.6.1-16-2016-00006 "The development and enhancement of the research potential at John von Neumann University" project. The Project is supported by the Hungarian Government and co-financed by the European Social Fund.

References

[1] KDD Cup 1999 Data http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[2] Ősz R., Holik I.: P*edagógiai kutatásmódszertan*, Óbudai Egyetem, 2015.

[3] Géron A.: *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow. Concepts, Tools, and Techniques to Build Intelligent Systems*, O'Reilly, 2019. ISBN 978-1-492-03264-9

[4] NSL-KDD dataset, https://www.unb.ca/cic/datasets/nsl.html

[5] Mitchell T. M.: *Machine learning*. McGraw-Hill Science/Engineering/Math, 1997.