***TASK 12***

***convert a non linear equation to linear one:***

***a linear equation is Y=m.X+c***

***Y,X=variable***

***m,c= constant***

***we just simplify the non \_linear equation to the form of a linear one***

***example:***

***y=a/x^2+b***

***y=a(1/x^2)+c***

***Y=y***

***X=1/x^2***

***m=a***

***c=c***

***so the x axis has the value of 1/x^2***

***WHAT IS PCA?***

**Principal component analysis, or PCA, is a statistical procedure that allows you to summarize the information content in large data tables by means of a smaller set of “summary indices” that can be more easily visualized and analyzed. The underlying data can be measurements describing properties of production samples, chemical compounds or reactions, process time points of a continuous process, batches from a batch process, biological individuals or trials of a DOE-protocol, for example.**

***Using PCA can help identify correlations between data points, such as whether there is a correlation between consumption of foods like frozen fish and crisp bread in Nordic countries.***

**Principal component analysis today is one of the most popular multivariate statistical techniques. It has been widely used in the areas of pattern recognition and signal processing and is a statistical method under the broad title of factor analysis.**

**PCA is the mother method for MVDA**

**PCA forms the basis of multivariate data analysis based on projection methods. The most important use of PCA is to represent a multivariate data table as smaller set of variables (summary indices) in order to observe trends, jumps, clusters and outliers. This overview may uncover the relationships between observations and variables, and among the variables.**

**PCA goes back to Cauchy but was first formulated in statistics by Pearson, who described the analysis as finding “*lines and planes of closest fit to systems of points in space”* [**[**Jackson, 1991**](https://pdfs.semanticscholar.org/70a9/769e8027b48bce5a9ce3a82d0ce88e0f0730.pdf)**].**

**PCA is a very flexible tool and allows analysis of datasets that may contain, for example, multicollinearity, missing values, categorical data, and imprecise measurements. The goal is to extract the important information from the data and to express this information as a set of summary indices called principal components.**

**Statistically, PCA finds lines, planes and hyper-planes in the K-dimensional space that approximate the data as well as possible in the least squares sense. A line or plane that is the least squares approximation of a set of data points makes the variance of the coordinates on the line or plane as large as possible.**

***PCA creates a visualization of data that minimizes residual variance in the least squares sense and maximizes the variance of the projection coordinates.***

**How PCA works**

**In a previous article, we explained why pre-treating data for PCA is necessary. Now, let’s take a look at how PCA works, using a geometrical approach.**

**Consider a matrix X with N rows (aka "observations") and K columns (aka "variables"). For this matrix, we construct a variable space with as many dimensions as there are variables (see figure below). Each variable represents one coordinate axis. For each variable, the length has been standardized according to a scaling criterion, normally by scaling to unit variance. You can find more details on scaling to unit variance in the previous blog post.**

***A K-dimensional variable space. For simplicity, only three variables axes are displayed. The “length” of each coordinate axis has been standardized according to a specific criterion, usually unit variance scaling.***

**In the next step, each observation (row) of the X-matrix is placed in the K-dimensional variable space. Consequently, the rows in the data table form a swarm of points in this space.**

***The observations (rows) in the data matrix X can be understood as a swarm of points in the variable space (K-space).***

**Mean centering**

**Next, mean-centering involves the subtraction of the variable averages from the data. The vector of averages corresponds to a point in the K-space.**

***In the mean-centering procedure, you first compute the variable averages. This vector of averages is interpretable as a point (here in red) in space. The point is situated in the middle of the point swarm (at the center of gravity).***

**The subtraction of the averages from the data corresponds to a re-positioning of the coordinate system, such that the average point now is the origin.**

***The mean-centering procedure corresponds to moving the origin of the coordinate system to coincide with the average point (here in red).***

**The first principal component**

**After mean-centering and scaling to unit variance, the data set is ready for computation of the first summary index, the first principal component (PC1). This component is the line in the K-dimensional variable space that best approximates the data in the least squares sense. This line goes through the average point. Each observation (yellow dot) may now be projected onto this line in order to get a coordinate value along the PC-line. This new coordinate value is also known as the *score*.**

***The first principal component (PC1) is the line that best accounts for the shape of the point swarm. It represents the maximum variance direction in the data. Each observation (yellow dot) may be projected onto this line in order to get a coordinate value along the PC-line. This value is known as a score.***

**The second principal component**

**Usually, one summary index or principal component is insufficient to model the systematic variation of a data set. Thus, a second summary index – a second principal component (PC2) – is calculated. The second PC is also represented by a line in the K-dimensional variable space, which is orthogonal to the first PC. This line also passes through the average point, and improves the approximation of the X-data as much as possible.**

***The second principal component (PC2) is oriented such that it reflects the second largest source of variation in the data while being orthogonal to the first PC. PC2 also passes through the average point.***

**Two principal components define a model plane**

**When two principal components have been derived, they together define a place, a window into the K-dimensional variable space. By projecting all the observations onto the low-dimensional sub-space and plotting the results, it is possible to visualize the structure of the investigated data set. The coordinate values of the observations on this plane are called scores, and hence the plotting of such a projected configuration is known as a score plot.**

***Two PCs form a plane. This plane is a window into the multidimensional space, which can be visualized graphically. Each observation may be projected onto this plane, giving a score for each.***

**Modeling a Data Set**

**Now, let’s consider what this looks like using a data set of foods commonly consumed in different European countries. The figure below displays the score plot of the first two principal components. These scores are called t1 and t2. The score plot is a map of 16 countries. Countries close to each other have similar food consumption profiles, whereas those far from each other are dissimilar. The Nordic countries (Finland, Norway, Denmark and Sweden) are located together in the upper right-hand corner, thus representing a group of nations with some similarity in food consumption. Belgium and Germany are close to the center (origin) of the plot, which indicates they have average properties.**

***The PCA score plot of the first two PCs of a data set about food consumption profiles. This provides a map of how the countries relate to each other. The first component explains 32% of the variation, and the second component 19%. Colored by geographic location (latitude) of the respective capital city.***

**How to Interpret the Score Plot**

**In a PCA model with two components, that is, a plane in K-space, which variables (food provisions) are responsible for the patterns seen among the observations (countries)? We would like to know which variables are influential, and also how the variables are correlated. Such knowledge is given by the principal component loadings (graph below). These loading vectors are called p1 and p2.**

**The figure below displays the relationships between all 20 variables at the same time. Variables contributing similar information are grouped together, that is, they are correlated. Crisp bread (crips\_br) and frozen fish (Fro\_Fish) are examples of two variables that are positively correlated. When the numerical value of one variable increases or decreases, the numerical value of the other variable has a tendency to change in the same way.**

**When variables are negatively (“inversely”) correlated, they are positioned on opposite sides of the plot origin, in diagonally 0pposed quadrants. For instance, the variables garlic and sweetener are inversely correlated, meaning that when garlic increases, sweetener decreases, and vice versa.**

***PCA loading plot of the first two principal components (p2 vs p1) comparing foods consumed.***

***If two variables are positively correlated, when the numerical value of one variable increases or decreases, the numerical value of the other variable has a tendency to change in the same way.***

**Furthermore, the distance to the origin also conveys information. The further away from the plot origin a variable lies, the stronger the impact that variable has on the model. This means, for instance, that the variables crisp bread (Crisp\_br), frozen fish (Fro\_Fish), frozen vegetables (Fro\_Veg) and garlic (Garlic) separate the four Nordic countries from the others. The four Nordic countries are characterized as having high values (high consumption) of the former three provisions, and low consumption of garlic. Moreover, the model interpretation suggests that countries like Italy, Portugal, Spain and to some extent, Austria have high consumption of garlic, and low consumption of sweetener, tinned soup (Ti\_soup) and tinned fruit (Ti\_Fruit).**

**Geometrically, the principal component loadings express the orientation of the model plane in the K-dimensional variable space. The direction of PC1 in relation to the original variables is given by the cosine of the angles a1, a2, and a3. These values indicate how the original variables x1, x2,and x3 “load” into (meaning contribute to) PC1. Hence, they are called loadings.**

**The second set of loading coefficients expresses the direction of PC2 in relation to the original variables. Hence, given the two PCs and three original variables, six loading values (cosine of angles) are needed to specify how the model plane is positioned in the K-space.**

***The principal component loadings uncover how the PCA model plane is inserted in the variable space. The loadings are used for interpreting the meaning of the scores.* *ere is the list of the top 10 C++ frameworks for top 10 C++ libraries used in AI***

[***Machine learning***](https://www.analyticsinsight.net/algorithmia-using-machine-learning-for-quick-secure-and-effective-production/)***is about calculations, and libraries assist machine learning specialists and designers to play out the computational undertakings without repeating the perplexing lines of codes. It assists coders with running calculations rapidly.***[***C++***](https://www.analyticsinsight.net/10-highest-paying-programming-languages-for-better-fortune-and-future/)***is great for dynamic burden adjusting, versatile storing, and growing enormous big data frameworks, and libraries. With some of the unique advantages of C++ as a programming language, (including memory management, performance characteristics, and systems programming), it serves as one of the most efficient tools for developing fast scalable data science and big data libraries. Here are the top 10 C++ frameworks for***[***machine learning***](https://www.analyticsinsight.net/top-machine-learning-jobs-to-apply-this-october-weekend/)***.***

***TensorFlow***

***TensorFlow is a famous***[***deep learning***](https://www.analyticsinsight.net/top-free-deep-learning-courses-for-beginners-in-2021/)***library created by Google with its environment of devices, libraries, community resources for machine learning. This library has a complete, adaptable environment of devices, libraries, and local area assets that lets analysts and engineers construct and convey ML-fueled applications without any problem. Regardless of whether you’re a specialist or an amateur, TensorFlow is an end-to-end platform that makes it easy for you to build and deploy ML models.***

**Caffe from Berkeley**

***Convolutional architecture for fast feature embedding or Caffe is written in C++ for a deep learning structure, has been created by the Berkeley Vision and Learning Center. The provisions of this library incorporate expressive engineering, extensible code, speed, and a huge local area which fosters dynamic advancement in exploration and industry arrangements.***

***Microsoft Cognitive Toolkit (CNTK)***

***Written in C++, Microsoft Cognitive Toolkit is a brought-together deep learning tool stash that depicts neural networks as a progression of computational advances through a coordinated chart. It carries out stochastic inclination plunge (SGD, mistake backpropagation) learning with programmed separation and parallelization across various GPUs and servers. CNTK permits clients to effortlessly acknowledge and join famous model sorts like feed-forward DNNs, convolutional nets (CNNs), and recurrent networks (RNNs/LSTMs).***

***mlpack Library***

***mlpack is a fast, flexible machine learning library, written in C++. The library aims to provide fast, extensible implementations of cutting-edge machine learning algorithms. It also provides simple command-line programs, Python bindings, Julia bindings, and C++ classes which can be integrated into larger-scale machine learning solutions.***

***SHARK Library***

***Shark is a quick, particular, general open-source machine learning library (C/C++), for applications and examination, with help for direct and nonlinear advancement, portion-based learning calculations, neural organizations, and different other machine learning strategies.***

***Armadillo***

***Armadillo is a direct polynomial math (C/C++) library with functionality similar to Matlab. The library is renowned for the fast transformation of exploration code into creation conditions, for design acknowledgment, PC vision, signal handling, bioinformatics, insights, econometrics, among others.***

***Faisis***

***This library (C/C++) is used for efficient similarity search and clustering of dense vectors. It contains algorithms that search in sets of vectors of any size, up to ones that possibly do not fit in RAM. It also has support for optional GPU provided via CUDA, and an optional Python interface.***

***OpenNN***

***Written in C++, open neural networks (OpenNN) is an open-source neural networks library for advanced analytics. The library contains sophisticated algorithms and utilities to deal with the following artificial intelligence solutions such as classification, regression, forecasting, among others. The main advantage of this library is its high performance.***

***FANN***

***Fast artificial neural network (FANN) is an open-source neural network library written in C language. The library implements multilayer artificial neural networks in C with support for both fully connected and sparsely connected networks. It is easy to use, versatile, well documented, and fast. The features include backpropagation training, evolving topology training, cross-platform, and can use both floating-point and fixed-point numbers.***

***Boosting***

***XGBoost – A parallelized optimized general purpose gradient boosting library.***

***ThunderGBM – A fast library for GBDTs and Random Forests on GPUs.***

***LightGBM – Microsoft’s fast, distributed, high-performance gradient boosting (GBDT, GBRT, GBM, or MART) framework based on decision tree algorithms, used for ranking, classification, and many other machine learning tasks.***

***CatBoost – General purpose gradient boosting on decision trees library with categorical features support out of the box. It is easy to install, contains fast inference implementation, and supports CPU and GPU (even multi-GPU) computation.***

* [**Exhibition | Conferences**](https://www.sartorius.com/en/company/exhibition-conferences)
* [**Contact**](https://www.sartorius.com/en/general-contact)

**© 2022 Sartorius AG**