**How do you convert nonlinear to linear?**

How would you go about converting 1y=x2+ab−x1y=x2+ab−x into linear form. I know how you would normally go about solving this type of problem but I fail to make any progress on this one.

I have to make it in the form Y=mX+cY=mX+c

where, YY and XX are in terms of xx and yy (the variables) without a and b (the constants); m and c are in terms of a and b (the constants) without x and y (the variables).

I'm having trouble separating the aa and xx in the ab−xab−x term.

Ok I was able to solve it, in case anyone was wondering...

1y=x2+ab−x1y=x2+ab−x

1y−x2=ab−x1y−x2=ab−x

log(1y−x2)=log(ab−x)log⁡(1y−x2)=log⁡(ab−x)

log(1y−x2)=(b−x)log(a)log⁡(1y−x2)=(b−x)log⁡(a)

log(1y−x2)=−log(a)×x+logablog⁡(1y−x2)=−log⁡(a)×x+log⁡ab

Y=mX+cY=mX+c

where Y=log(1y−x2)Y=log⁡(1y−x2) , m=−log(a)m=−log⁡(a) , X=xX=x and c=logab

**What is PCA and how it works?**

The main idea of [principal component analysis](https://www.projectpro.io/recipes/what-is-principal-component-analysis-statsmodels-library) (PCA) is to reduce the dimensionality of a data set consisting of many variables correlated with each other, either heavily or lightly, while retaining the variation present in the dataset, up to the maximum extent. The same is done by transforming the variables to a new set of variables, which are known as the principal components (or simply, the PCs) and are orthogonal, ordered such that the retention of variation present in the original variables decreases as we move down in the order. So, in this way, the 1st principal component retains maximum variation that was present in the original components. The principal components are the eigenvectors of a covariance matrix, and hence they are orthogonal.

Importantly, the [dataset](https://www.projectpro.io/article/100-machine-learning-datasets-curated-for-you/407) on which PCA technique is to be used must be scaled. The results are also sensitive to the relative scaling. As a layman, it is a method of summarizing data. Imagine some wine bottles on a dining table. Each wine is described by its attributes like colour, strength, age, etc. But redundancy will arise because many of them will measure related properties. So what PCA will do in this case is summarize each wine in the stock with less characteristics.

Intuitively, Principal Component Analysis can supply the user with a lower-dimensional picture, a projection or "shadow" of this object when viewed from its most informative viewpoint.

Image Source: Machine Learning Lectures by Prof. Andrew NG at Stanford University:

Dimensionality : It is the number of random variables in a dataset or simply the number of features, or rather more simply, the number of columns present in your dataset.

Correlation : It shows how strongly two variable are related to each other. The value of the same ranges for -1 to +1. Positive indicates that when one variable increases, the other increases as well, while negative indicates the other decreases on increasing the former. And the modulus value of indicates the strength of relation.

Orthogonal: Uncorrelated to each other, i.e., correlation between any pair of variables is 0.

Eigenvectors: Eigenvectors and Eigenvalues are in itself a big domain, let’s restrict ourselves to the knowledge of the same which we would require here. So, consider a non-zero vector v. It is an eigenvector of a square matrix A, if Av is a scalar multiple of v. Or simply:

Av = ƛv

Here, v is the eigenvector and ƛ is the eigenvalue associated with it.

Covariance Matrix: This matrix consists of the covariances between the pairs of variables. The (i,j)th element is the covariance between i-th and j-th variable.

**What are C++ Libraries / Frameworks for AI?**

**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.