Introduction

In this write up, we present a unified picture to connect many machine learning algorithms. The critical point in this picture is the so-called basis change. Normally the basis change refers to linear transforming one basis vectors from one space to another. Here however, we also extend this to the nonlinear basis change. In the context of neural network, it is often straightforward to generate a nonlinear basis representation (e.g. with auto-encoder), without knowing any specific closed form of the corresponding nonlinear operator.

Although sometimes a basis change can directly give the solution to a machine learning problem, e.g., in the linear regression, the general steps for solving many machine learning problems are:

- Transform the original data to a new basis.
- Find the solution (parameters of the model) by optimizing a certain quantity on that new basis.

The purpose of using the new basis might be because it is simpler or more proper to handle the relevant quantity, or because we can work in a very low dimension within that basis. As can be seen later, finding the most relevant but lowest dimension representation of original data should be a priority in solving machine learning problems. This will not only make the relevant problem tractable in calculation, but also make it less prone to overfitting. The use of filters in convolutional neural network (CNN), the window functions in recurrent neural network (RNN), etc. are all meant to transform high-dimensional vector to a low-dimension one. Dimension reductions through either simple feature selection methods or complicated linear / nonlinear basis change, are very crucial in many machine learning applications. Sometimes, they make the problem solving from impossible to possible, or at least from intractable to tractable.

Next, we will briefly describe examples on how machine algorithms employ basis change to facilitate the problem solving process. Algorithms using linear basis change will be first introduced, and after that, the algorithms based on nonlinear basis change.

Linear basis change

For each algorithm, we will following the two steps of using basis change as described earlier.

PCA

- Projecting the random vector $\mathbf{x}$ to a basis vector u of another basis.
- Within the new basis $\{u\}$, maximize the variance.

The variance along u is

$$E[(\mathbf{x} \cdot u - E[\mathbf{x} \cdot u])^2] = E[(u \cdot (\mathbf{x} - E[\mathbf{x}]))^2] = u^T E[(\mathbf{x} - E[\mathbf{x}]) \cdot (\mathbf{x} - E[\mathbf{x}])^T] u = u^T C u$$

Maximizing the variance subject to the condition $||u||_2^2 = 1$ (otherwise variance is arbitrary as u scales) turns out to be finding the eigenvectors of covariance matrix C.

Details are in "principal component analysis_SVD.pdf" under /algorithm folder.

LDA

- Projecting the random vector $\mathbf{x}$ to a basis vector v of another basis.
- Within the basis $\{v\}$, maximize the Fisher's discriminant
$$J(v) = \frac{(\tilde{\mu_1} - \tilde{\mu_2})^2}{\tilde{s_1}^2 + \tilde{s_2}^2} = \frac{v^T S_B v}{v^T S_w v}$$
Maximizing the Fisher discriminant turns out to be solving a generalized eigenvalue problem and find the new basis $\{v\}$.

Details are in "linear discriminant analysis_compared to PCA.pdf" under /algorithm folder.

ICA

- Projecting the random vector $\mathbf{x}$ to a basis vector w of another basis.
- Within the basis $\{w\}$, maximize the Non-Gaussianity, usually described by, e.g., kurtosis.

Details are in "independent component analysis.pdf" under /algorithm folder.

Linear regression

- $y = \theta^T x = w^T x + b$ projecting a random vector x to another basis (usually 1D).
- Within the transformed basis, minimizing squared error to determine the parameters. Here, because we usually perform a linear transformation from multiple-dimension to one dimension, the corresponding matrix for this transformation is usually a row vector.

In the typical example of linear regression shown in a two-dimensional $x - y$ graph, the vector x becomes a 1D scaler, and the linear transformation has the form $y = \beta x + b$, where the 'linear transformation matrix' is just a number β.

Details of other interpretations of linear regression are in the file 'linear regression theory_SVD_projection to column space_basis change_gradient descent.pdf' under the folder of algorithm.

Logistic regression

- $y = \frac{1}{1+e^{-\theta^T x}}$ projecting a random vector x to another basis (usually 1D dimension with multiple categorical values).
- Within the transformed basis, minimizing cross-entropy or maximizing likelihood to find the parameters.

Though with similarity to linear regression, the basis change with sigmoid function above is obviously not a linear transformation. However, logistic regression is a generalized linear model. Details can be found in "generative vs discriminative algorithms_GLM_GDA.pdf" under the /fundamentals folder.

In logistic regression, we assign a class when the y value is above, e.g. 0.5. However, this is equivalent to considering $z = \theta^T x \geq 0$. Thus, although being implemented by a nonlinear transformation, logistic regression is still a linear classifier, as $z = \theta^T x \geq 0$ gives a linear decision boundary.

SVM

This section need further verification.

- The key point is projecting data to a new vector w.
- Maximizing the geometric margin along the new vector w.

SVM without kernel should correspond to a linear basis change, while SVM with kernel should correspond to a nonlinear basis change.

See details in "SVM_theory.pdf" under /algorithm folder.

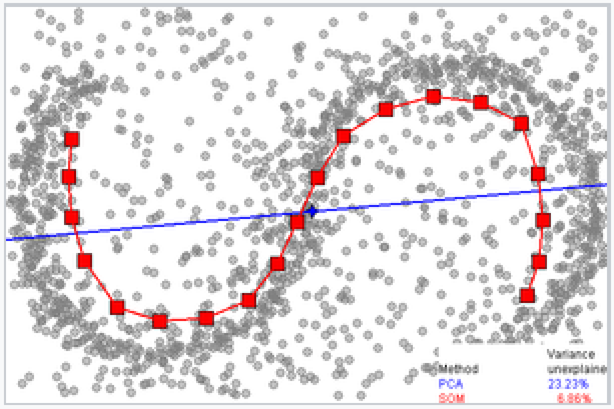
Summary

In the algorithms introduced above, we are solving problems by following a process of 'projecting to a new basis, on which optimize a specific quantity for that problem'.

Nonlinear basis change

Introduction

Nonlinear basis change is much less common than the above introduced linear basis change, and generally considered not so useful https://math.stackexchange.com/questions/1096325/change-of-basis-with-a-nonlinear-operator (https://math.stackexchange.com/questions/1096325/change-of-basis-with-a-nonlinear-operator). This is probably due to the fact that representing data with explicit nonlinear relationship is almost impossible, as there are infinite possible nonlinear relations among data. However, in machine learning, particularly with the deep neural networks, one can train the data to make use of the nonlinear structures among data without knowing their explicit nonlinear dependence. Below is an example to show the importance of nonlinear representation https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction (https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction).



Approximation of a principal curve by one-dimensional SOM (a broken line with red squares, 20 nodes). The first principal component is presented by a blue straight line. Data points are the small grey circles. For PCA, the Fraction of variance unexplained in this example is 23.23%, for SOM it is 6.86%.[17]

As shown in the figure above, the data has strong nonlinear structures among their different features / dimensions. If we use only linear PCA and project the data to a line, then we cannot represent the data very well along any directions. If we use the coordinates projected to an optimal axis obtained by PCA and feed them to a machine learning algorithm, then for sure we cannot get good results. However, a nonlinear representation shown in the figure achieves a much better result.

Manifold learning algorithms

Most nonlinear mapping algorithms belong to the category of manifold learning, where a lower-dimensional manifold embedded in a much higher dimension space is learned. https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction (https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction). There are many ways to realize a nonlinear mapping such as using probabilistic models to minimize KL divergence, neural network training, or the combination of the two, etc. The typical goal of such nonlinear basis change (mapping) is to represent 'almost correctly' with a much lower dimension vector. The critical point point making this possible is there are often only a small number of 'intrinsic' dimensions in a higher dimension data, which can be revealed by nonlinear mapping. Below are some examples.

- Face encoding algorithm using CNN maps a high-dimensional vector to a dense but representative low-dimensional vector, which significantly facilitate applications such as face recognition, image searching etc.
- The family of auto-encoder algorithms are more typical examples to do nonlinear basis change in order to achieve a dimension reduction.
- t_SNE maps high dimensional data to an even lower two to three dimensions data for visualization, or for further processing by other classification algorithm.
- Word embeddings can also be regarded as a non-linear mapping to improve the quality of natural language processing.

Although the ways to implement a nonlinear basis change varies significantly, it is still possible to unify them with the same steps as in the case of linear basis change. That is, **project the data to a new basis, and then solve the problem by optimizing a specific objective function relevant to that problem**. For example, in the case of neural network, the number of nodes in hidden layer can be regarded as the dimension of a nonlinear basis, while the coefficients after activation can be treated as the coordinates on the space with this dimension. The coordinates or coefficients can be solved by optimizing a specific objective function.

Some more details of the examples of nonlinear basis change or nonlinear dimension reduction can be found in the "nonlinear dimension reduction _ NLDR.pdf" under the /algorithm folder.

Summary

We present a tentative picture to unify many machine learning algorithms in terms of either linear or nonlinear basis change. A basis change is often related to a dimension reduction, which is particularly true in the case of high dimensional data such as image, languages, etc. Moreover, because dimension reduction is often related to the effort of preventing overfitting (such as in the cases of CNN, RNN etc.), studying linear or nonlinear basis change is thus still one of efforts to handle the fundamentals of machine learning: which is bias and variance.