Sparse online supervised PCA

Nathanael Fillmore
Computer Sciences Department
University of Wisconsin-Madison
nathanae@cs.wisc.edu
May 14, 2009

1 Introduction

For some machine learning problems the number of possible variables is extremely large. Consider the following two examples:

Example #1. In text processing, any of the following could be used as variables:

- 1-gram counts, 2-gram counts, ..., N-gram counts, for any N;
- exact position indicators, i.e., the variable $X_{w,i}$ can be defined to be 1 if word type w occurs at position i in a document, and 0 otherwise;
- punctuated N-gram counts, i.e., for each index set $\mathcal{S} := \{s_1, s_2, \dots\} \subset \{1, 2, \dots\}$, and for each combination of word types $w_1, w_2, \dots, w_{|\mathcal{S}|}$, the variable $X_{\mathcal{S},w_0,w_1,\dots,w_{|\mathcal{S}|}}$ can be defined as the number of times word type w_1 occurs s_1 positions before word type w_0, word type w_2 occurs s_2 positions before w_0, ..., and word type $w_{|\mathcal{S}|}$ occurs $s_{|\mathcal{S}|}$ positions before w_0, in a document;

and so on. The number of possible combinations is unbounded.

Example #2. A robot may have the following capabilities:

- it can take photos arbitrarily frequently, in arbitrary directions, capturing light at arbitrary wavelengths;
- it can record sounds, temperatures, motions, etc., in similarly diverse ways;
- it can move itself and it can push or pull other objects, and afterwards it can measure the effects of its actions.

As in the previous example, these capabilities lead to an unlimited number of possible variables that the robot can, in principle, observe.

However, although the number of possible variables may be unbounded, the number of variables it is practical to observe is always limited. This limitation has at least two causes.

First, as is well known, *learning* becomes increasingly difficult as the number of variables increases, due to Bellman's "curse of dimensionality" [2] and related issues; see [3] for an overview. For example, it is a standard result of computational learning theory that learning

an unbiased estimator in p-dimensional boolean space has sample complexity exponential in p [7].

Second, the *computational complexity* associated with a large number of variables is sometimes great. For example, when the number of variables $p = 100\,000$, the covariance matrix contains $p(p+1)/2 \approx 5$ billion unique entries. Note that this figure is independent of n, the number of training examples: even if n is 1, computation of the covariance matrix will require time quadratic in the number of variables, and—sometimes even worse—will also require space quadratic in the number of variables.

Of course, it is clear that not all variables are equally important for a given problem. Thus a large number of feature selection techniques have been devised to reduce the number of variables; see [5] for an overview, and below for some specific techniques.

Many feature selection techniques address only the first problem mentioned above, that of learning, but do not address the computational complexity problem. Indeed, use of feature selection can require more computation than learning directly from the original variables.

In this paper, we present a feature selection algorithm, SOSPCA, that addresses both the learning problem and the computational problem. The algorithm can select a small, transformed subset of variables from arbitrarily many original variables, in time linear in the number of variables in the worst case, and in expected constant space. The algorithm has these properties because it is an "online" algorithm, not in the usual sense that it considers a stream of observations, but rather in the sense that it considers a stream of variables.

2 Problem definition and algorithm

2.1 Problem definition

We consider p input variables $X_1, X_2, \ldots, X_p$ and one output variable Y. We assume for convenience that all variables are quantitative and are centered on zero, but neither assumption is necessary. The number p may be very large.

During training, we make n observations of the output variable and of each input variable. In contrast to usual practice, we do not assume that all variables are observed at once, but we do assume that all observations of a particular variable are made at once. (For example, we may not count all N-grams in a corpus up front, but when we consider one particular N-gram, we do count it in all documents.) Our notation reflects this fact: we array the n observations of the variable X_i in an n-dimensional vector $x_i := [x_{1i}\ x_{2i}\ \cdots\ x_{ni}]^T$, and the n observations of the variable Y in an n-dimensional vector $y := [y_1\ y_2\ \cdots\ y_n]^T$.

We seek a set of $q \ll p$ linearly transformed input variables $Z_1, Z_2, \ldots, Z_q$ that captures the correlation observed during training between the original p input variables and the output variable. We will end up with a linear change-of-basis operator $B \in \mathbb{R}^{p \times q}$ that maps each

observation from its representation under the p original variables to its representation under the q transformed variables.

We always use i as an index for original variables ($i = 1, \ldots, p$), j as an index for observations ($j = 1, \ldots, n$), and k as an index for transformed variables ($k = 1, \ldots, q$).

2.2 Algorithm

At a high level, the SOSPCA algorithm is as follows:

- Begin with q arbitrary "transformed" variables, represented in the standard q-dimensional basis.
- For each remaining original variable:
  - If the original variable is more highly correlated with the output variable than one of the transformed variables is correlated with the output variable:
    - Add the original variable as a transformed variable.
    - Merge the two transformed variables that are least correlated with the output variable.

In more detail, the SOSPCA algorithm is as follows:

- Initially:
  - Let $Z_1 = X_1, Z_2 = X_2, \ldots, Z_q = X_q$, and let $z_1 = x_1, z_2 = x_2, \ldots, z_q = x_q$.
  - Let $B := [b_1\ b_2\ \cdots\ b_q]$, the change of basis operator, be the first q columns of the $p \times p$ identity matrix.
  - Compute the sample correlation

$$\rho_{kY} = \frac{z_k^T y}{\sqrt{z_k^T z_k}\sqrt{y^T y}}$$

    between each of the variables Z_k, $k = 1, \ldots, q$, and the output variable Y.
- For $i = q+1, q+2, ..., p$:
  - Compute the sample correlation ρ_{iY} between the variable X_i and Y as above (but substituting x_i for z_k).
  - If $\rho_{iY} > \min_{k=1,\ldots,q}\ \rho_{kY}$:
    - Temporarily let $Z_{q+1} = X_i$, $z_{q+1} = x_i$, and $B = [B \mid e_i]$, where e_i is the ith standard basis vector, with a 1 in the ith place and 0s elsewhere.
    - Let $k' = \arg\min_{k=1,\ldots,q+1} \rho_{kY}$, and let $k'' = \arg\min_{k=1,\ldots,q+1,\ k\neq k'} \rho_{kY}$ be the indices of the two variables $Z_{k'}$ and $Z_{k''}$ that are least highly correlated with Y.
    - Let $\sigma^2_{k',k''}$ be the sample covariance between $Z_{k'}$ and $Z_{k''}$, and let $\sigma^2_{k',k'}$ be the sample variance.
    - Let $\theta = (1/2)\tan^{-1}[(2\sigma_{k',k''})/(\sigma_{k',k'}\sigma_{k'',k''})]$ be the angle between $z_{k'}$ and $z_{i''}$.

- Update

$$\left(b_{k'} \mid b_{k''}\right) = \left(b_{k'} \mid b_{k''}\right) \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}, \quad \text{and}$$
$$\left(z_{k'} \mid z_{k''}\right) = \left(z_{k'} \mid z_{k''}\right) \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix},$$

  thus rotating the basis so as to decorrelate $z_{z'}$ and $z_{z''}$.
  - Recompute $\rho_{k'Y}$ and $\rho_{k''Y}$.
  - If $\rho_{k'Y} > \rho_{k''Y}$, discard $Z_{k''}$, by shifting $Z_k = Z_{k+1}$ and $z_k = z_{k+1}$ for $k = k'+1, \ldots, q$ and by updating $B = [b_1 \ \cdots \ b_{k'-1} \ b_{k'+1} \ \cdots \ b_{q+1}]$.
  - Otherwise, discard $Z_{k'}$ similarly.
- Otherwise:
  - Ignore variable X_i.

2.3 Example

To illustrate with the mechanics of the algorithm and its attractive properties, we consider the following example. In the following, we denote $\mathbf{X} = [x_1 \ x_2 \ \cdots \ x_p]$ and $\mathbf{Z} = [z_1 \ z_2 \ \cdots \ z_q]$.

In our example, we are given $p = 5$ input variables, one output variable, and $n = 3$ observations of the variables, as follows:

$$\mathbf{X} = \begin{pmatrix} 2.0000 & 1.6667 & 1.6667 & -0.6667 & 2.6667 \\ 2.0000 & -0.3333 & 2.6667 & -1.6667 & -2.3333 \\ -4.0000 & -1.3333 & -4.3333 & 2.3333 & -0.3333 \end{pmatrix}, y = \begin{pmatrix} 5.3333 \\ -4.6667 \\ -0.6667 \end{pmatrix}.$$

We seek to extract $q = 2$ variables.

We initialize $\mathbf{Z}$ as $[x_1 \ x_2]$:

$$\mathbf{Z} = \begin{pmatrix} 2.0000 & 1.6667 \\ 2.0000 & -0.3333 \\ -4.0000 & -1.3333 \end{pmatrix},$$

and the basis to the standard basis:

$$B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

This basis reflects the fact that so far we have simply chosen the first two variables. The final preliminary step is to compute the correlation of each transformed variable Z_k with the output variable Y:

$$\rho_{k=1,Y} = 0.1147, \rho_{k=2,Y} = 0.7370.$$

Next, we enter the first iteration of the main loop. We compute the sample correlation between X_i and Y for $i = 3$. This quantity is $\rho_{i=3,Y} = -0.0175$, which is less than both $\rho_{k=1,Y} = 0.1147$ and $\rho_{k=2,Y} = 0.7370$. Thus, we ignore variable X_3.

Next, we enter the second iteration of the main loop. We compute the sample correlation between X_i and Y for $i = 4$. This quantity is $\rho_{i=4,Y} = 0.1273$, which is larger than $\rho_{k=1,Y} = 0.1147$. Thus we cannot simply throw away X_4. Instead, we temporarily add X_4 to the set of transformed variables, under the alias Z_3, and we add e_4 to the basis B:

$$\mathbf{Z} = \begin{pmatrix} 2.0000 & 1.6667 & -0.6667 \\ 2.0000 & -0.3333 & -1.6667 \\ -4.0000 & -1.3333 & 2.3333 \end{pmatrix}, B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}.$$

Of Z_1, Z_2, and Z_3, the variables Z_1 and $Z_3(= X_4)$ are least highly correlated with Y (see above). Thus we rotate the basis B in dimensions 1 and 3 so as to decorrelate Z_1 and Z_3. After the rotation, we end up with

$$\mathbf{Z} = \begin{pmatrix} 2.0547 & 1.6667 & 0.4721 \\ 2.5753 & -0.3333 & -0.3817 \\ -4.6299 & -1.3333 & -0.0903 \end{pmatrix}, B = \begin{pmatrix} 0.8538 & 0 & 0.5206 \\ 0 & 1.0000 & 0 \\ 0 & 0 & 0 \\ -0.5206 & 0 & 0.8538 \\ 0 & 0 & 0 \end{pmatrix}.$$

The new correlation of Z_1 with Y is $\rho_{k=1,Y} = 0.050110$, while the new correlation of Z_3 with Y is $\rho_{k=3,Y} = 0.997806$. The new variable Z_3 is almost parallel to Y, a fact that is highly useful for future learning. In contrast, Z_1 is less highly correlated with Y, so we discard Z_1, and the final representations of our data and the basis after the second iteration are:

$$\mathbf{Z} = \begin{pmatrix} 1.6667 & 0.4721 \\ -0.3333 & -0.3817 \\ -1.3333 & -0.0903 \end{pmatrix}, B = \begin{pmatrix} 0 & 0.5206 \\ 1.0000 & 0 \\ 0 & 0 \\ 0 & 0.8538 \\ 0 & 0 \end{pmatrix}.$$

We can continue this process as long as more variables X_i are available; in the present example one more iteration will be taken.

3 Theory

Recall that we assume that $p \gg q$; we also assume $p \gg n$.

3.1 Time

The SOSPCA algorithm requires time only linear in the number p of original variables. The analysis is as follows. Each step of our algorithm requires constant time with respect to the

number of original variables. At each step, we must compute the sample correlation between the variable currently under consideration and the target variable; this requires $O(n)$ time. If this correlation is small, we immediatly go to the next step. Otherwise, we need to perform more computations, but only at most $O(\min(qn^2, nq^2))$ of them, to compute the sample covariance between the n variables. Thus, the total runing time is $O(p \min(qn^2, nq^2))$.

3.2 Space

In the worst case, the SOSPCA algorithm also requires space linear in the number of original variables, in order to store the change-of-basis operator B. However, under certain assumptions, only space constant in the number of original variables is required.

The analysis is as follows. Rows in B that correspond to

- variables that have not yet been considered, i.e., $X_{i'}, i' > i$, and
- variables that have been considered but immediately discarded because they have low correlation with the output variable

do not need to be stored in B. In [4], the following theorem is shown:

Theorem 1. *If t correlation variables ρ_t are drawn iid from a uniform distribution over the interval $[-1, 1]$, then the expected number of times $\rho_t > \max_{t'=1,\ldots,t-1} \rho_{t'}$ is asymptotic to $(\log t)/t$.*

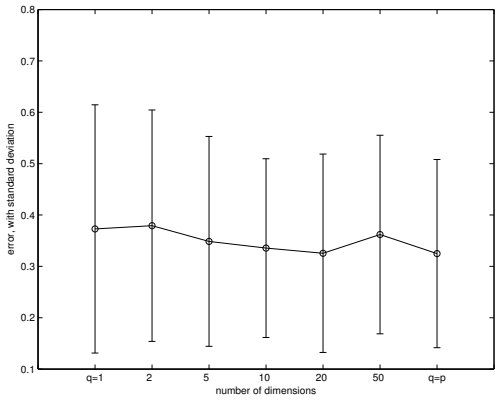
As an immediate consequence of this theorem, the expected number of rows required in B is proportional to $(\log p)/p \to 0$ as $p \to \infty$.

4 Experimental evaluation

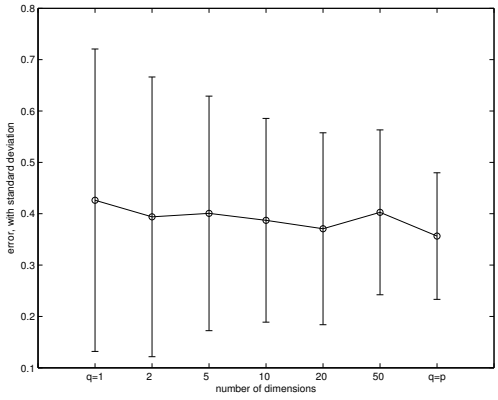
In this section we present an experimental validation of the above results. The experiments are not conclusive, but they do show that using our algorithm, dimensionality can be greatly reduced without increasing average error by a large amount, and that in some cases error is reduced by a significant amount.

Our methodology is as follows. We collect ten datasets prepared by other students in the University of Wisconsin's machine learning course. In all cases, the output variable is binary $\{-1, 1\}$. Some input variables are quantitative and some are qualitative. Each K-valued qualitative variable was replaced by K binary $\{-1, 1\}$ indicator variables. The quantitative variables were rescaled into $[-1, 1]$ to avoid ill-conditioning.

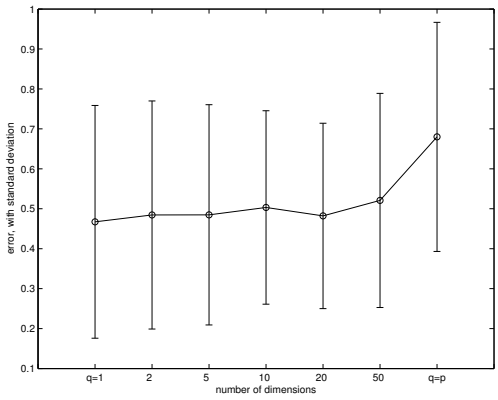
We consider three classification algorithms: (a) linear discriminant analysis, (b) support vector machine with a Gaussian kernel ($\sigma = 0.3$), and (c) 1-nearest neighbor. For each algorithm, we train using no feature selection, or feature selection with $p = 1, 2, 5, 10, 20$ and 50 variables using the SOSPCA algorithm. In all cases, we use 10-fold cross-validation on each of the datasets separately.



(a) LDA



(b) 1-NN



(c) SVM

Figure 1: Error plots. See text for details. $q = p$ denotes the case where no feature selection was performed.

Results are shown in Figure 1. Each plot corresponds to a distinct classification algorithm, and each abscissa corresponds to a distinct number of variables in the transformed representation. Average error accross folds and datasets is indicated by a circle, and the standard deviation is indicated by vertical bars.

For LDA and 1-NN, error increases somewhat as the number of variables selected decreases. However, for LDA none of the increases are significant at the 95% level, and for 1-NN, only the increases at $q = 1$ and $q = 50$ are significant. This result is somewhat encouraging, because it shows that the SOSPCA algorithm can reduce dimensionality from $p > 50$ to $q = 2$ without increasing error by a significant amount.

For SVM with a Gaussian kernel, the error actually decreases as the number of variables selected decreases. Indeed, the lowest average error was obtained for $q = 1$. All the decreases in error for SVM relative to the $q = p$ case are significant at the 95% level.

Why is the difference significant in the case of SVM? It seems that the reason is related to the use of a Gaussian kernel: with polynomial and linear kernels, the same effect was not observed in preliminary experiments. One possibility is as follows. Recall that for multivariate Gaussian density functions, the mean and covariance are sufficient statistics. Thus it seems reasonable to suppose that a Gaussian kernel will work best to the extent that the sample covariance of the input variables is related to the variability of the output variable. But this property is precisely what the SOSPCA algorithm aims to achieve. So it is reasonable that SOSPCA would be well-suited as a feature selection algorithm prior to using SVM with a Gaussian kernel. In contrast, LDA and 1-NN do not have this "sufficient statistic" property.

5 Related work

As mentioned in the introduction, a large number of feature selection algorithms have been proposed. We briefly mention two approaches that are most significant to our work: supervised principal components [1] and treelets [6]. In contrast to both these approaches, SOSPCA is online with respect to the variables. In contrast to treelets, SOSPCA is supervised: it selects variables to decorrelate based on their correlation to the target variable, whereas treelets selects variables based on their correlation to each other. In contrast to supervised principal components, SOSPCA incrementally performs local Jacobi rotations on pairs of variables; in contrast, supervised principal components chooses a subset of variables (based on correlation to the target variable) and then performs PCA on all these variables at once.

6 Future work

In the future, we would like, first, to explore further the theoretical properties of the SOSPCA algorithm. Is the SOSPCA an algorithm asymptotically unbiased estimator of something?

Does it reduce to another algorithm under certain conditions? And so on. Second, it is important to perform a much more thorough empirical validation. In particular, it is desirable (a) to compare the SOSPCA algorithm to a large number of competing approaches, and (b) to tune parameters of the classifiers more systematically, to see how this affects the value of the SOSPCA as a feature selection algorithm.

References

[1] E. Bair, T. Hastie, D. Paul, and R. Tibshirani. Prediction by supervised principal components. *Journal of the American Statistical Association*, 2006.

[2] R. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1961.

[3] Robert Clarke, Habtom W. Ressom, Antai Wang, Jianhua Xuan, Minetta C. Liu, Edmund A. Gehan, and Yue Wang. The properties of high-dimensional data spaces: implications for exploring gene and protein expression data. *Nature Reviews Cancer*, 2008.

[4] Nathanael Fillmore. Efficient computation of correlation matrices. *Course project for CS809, University of Wisconsin-Madison*, 2009. `http://pages.cs.wisc.edu/~nathanae/corr.pdf`.

[5] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[6] Ann B. Lee, Boaz Nadler, and Larry Wasserman. Treelets—an adaptive multi-scale basis for sparse unordered data. *The Annals of Applied Statistics*, 2(2):435–471, 2008.

[7] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.