Reconstructing Broadcasts on Trees

Douglas Stanford
CS 229
December 13, 2012

Abstract

Given a branching Markov process on a tree, the goal of the reconstruction (or "broadcast") problem is to guess the initial condition at the root, given the states of the leaves. In this essay, we evaluate the performance of some algorithms, new and old, for solving the reconstruction problem.

1 Introduction

Branching Markov processes have a wide relevance across disciplines. In phylogenetics, they arise as a description of the mutation dynamics of evolution. In mathematics and physics, they are simple examples of statistical mechanics systems with phase transitions. In theoretical cosmology, they are used as a model for the fluctuation dynamics of inflation.

In all three of these contexts, a natural and important question is the following: how well can one recover the initial state of the branching Markov process from the final state on the leaves of the tree? In this essay, we will apply machine learning algorithms to this question.

First, a definition. In this essay we will work with two-state branching Markov systems. This is defined on a regular rooted tree of degree d as follows. Begin by specifying the state of the root, at generation $u = 0$, as either white or black. Then, for each of the $(d-1)$ children of the root, independently assign the child the color of the parent with probability $(1-p)$ and the opposite color with probability p. Repeat this process recursively to color the entire tree.

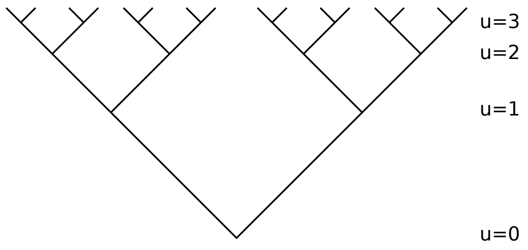

Figure 1: The time variable u in a $p = 2$ tree.

The reconstruction problem [1] asks the following question: given the colors of the leaves, with what probability can the initial state of the root be inferred? The answer depends on the value of u, the degree d and the probability of color flip p. It is known [2] that, in the limit of large u, the reconstruction probablem is unsolvable if

$$1 - 2p \geq \frac{1}{\sqrt{d-1}}. \tag{1}$$

However, if the reverse inequality holds, then the leaves contain at least some information about the state of root, even in the $u \to \infty$ limit [3, 4].

In this project, we will fix u and d and vary p. This gives us an environment in which the difficulty of the classification problem can be controlled precisely. We will study the performance of four different algorithms as a function of p: naive Bayes, not-so-naive Bayes, parsimony and an SVM.

2 The dataset

We generated training and testing sets by running the branching Markov process for different values of p. Sample configurations of the leaves for a degree five tree are shown in figure 2. These configurations all had white initial conditions at the root. The value of p increases from the left panel to the right panel to the bottom panel, and it is intuitively obvious that the reconstruction becomes more difficult as p increases.

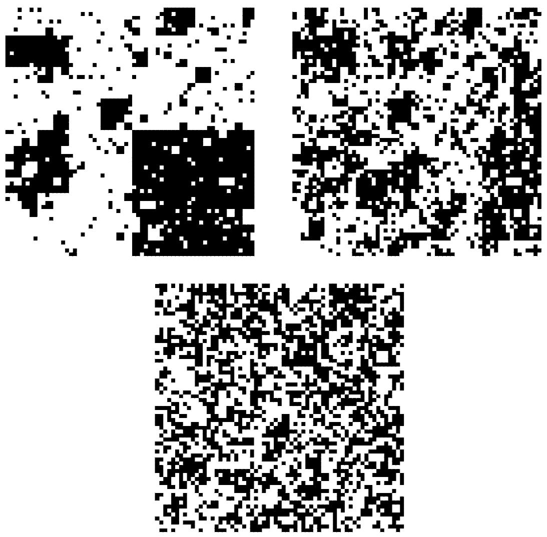

Figure 2: Sample configurations of the leaves in the $d = 5$ two-state system after six generations, for different values of p.

To get a feel for the important variables, we ran PCA on a set of 10000 training examples. The first principal component ends up being roughly constant on all leaves. Naively, this encodes most of the information about whether the root was white or black. The second, third and fourth principal components are shown in figure 2, again for a degree five tree. The four obvious blocks correspond to the descendants of the four children of the root. These principal components have an approximate interpretation as labelling whether one of the first children had a color change, compared to the root.

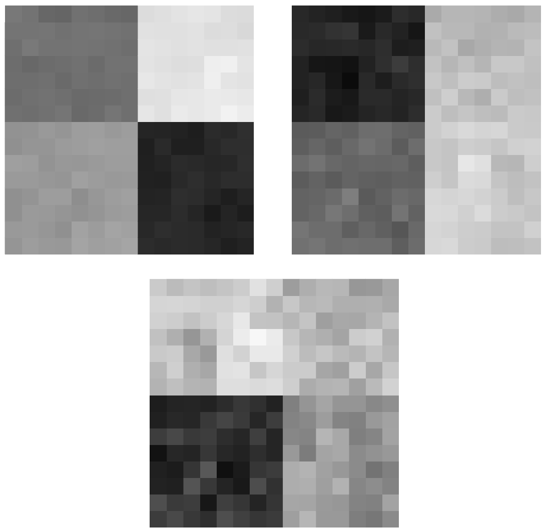

Figure 3: The second, third and fourth principal components for the $d = 5$ system. The first principal component is approximately spatially homogeneous.

Although the data shown here is for the $d = 5$ tree, it is convenient to run the algorithms (particularly parsimony) on a tree of even degree, so we chose to work with $d = 4$. We generated 20000 training and testing examples for each of twelve values of p in the range 0.01 to 0.23, with randomly chosen white/black initial conditions. For reference, the critical value beyond which reconstruction is asymptotically ($u \to \infty$) impossible is approximately $p = 0.21$ for a degree four tree.

3 Naive-ish Bayes

The optimal (maximum likelihood) reconstruction of the root would proceed as follows. We would assign a uniform prior for $P(root)$ and then compute

$$P(root|leaves) = \frac{P(leaves|root)P(root)}{P(leaves)}. \tag{2}$$

As usual $P(leaves)$ cancels out when comparing probabilities for classification. All we need is $P(leaves|root)$. In principle, explicit knowledge of the Markov matrix and the tree graph make is possible to compute this exactly. However, it appears to be computationally intractable.

One alternative is to use a maximum likelihood algorithm on a related but simpler graph. For example, choosing the graph on the left of figure 4 is equivalent to making an ansatz

$$P^{NB}(leaves|root) = \prod_i P(leaf_i|root). \tag{3}$$

Here, $P(leaf_i|root)$ can be computed simply by iterating the Markov matrix. Explicitly, it is

$$P(leaf_i|root) = \frac{1}{2} \pm \frac{1}{2}(1-2p)^u, \tag{4}$$

with the upper sign if the leaf is the same color as the root, and the lower sign if the leaf and root are opposite. After taking the product over leaves, it becomes clear that maximizing P^{NB} over choices of the root color reduces to majority vote: if most of the leaves are white, guess that the root was also white. Otherwise, guess that the root was black.

A slightly more complicated graph for which maximum likelihood is nevertheless tractable is shown on the right in figure 4.

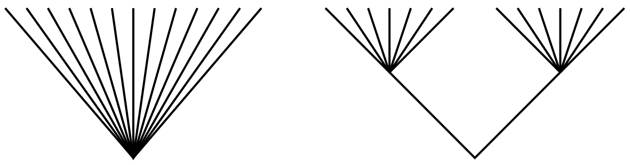
Figure 4: Simplified tree graphs for the naive Bayes model (left) and the NSNB model (right)

This is much like the naive Bayes graph on the left, but it models the first generation after the root explicitly as latent variables:

$$P^{NSNB}(leaves|root) = \sum_{\{z\}} P(\{z\}|root) \prod_i P(leaf_i|\{z\}). \tag{5}$$

Here, $\{z\}$ are latent varibles labeling the colors of first generation, $P(\{z\}|root)$ is equal to $p^m(1-p)^n$, where n is the number of vertices in the first generation with the same color as the root, and m is the number with the opposite color. Finally, $P(leaf_i|\{z\})$ is as in eq (4), but with the color of the root replaced by the color of the relevant first generation vertex, and u replaced by $u-1$.

In the text below, we refer to this model as the not-so-naive-Bayes (NSNB) algorithm.

4 Parsimony

The parsimony reconstruction algorithm, popular in the phylogenetic literature, recursively applies majority vote.[1] More specifically, the leaves at generation u are grouped into families of size $(d-1)$, each having a single parent at generation $(u-1)$. The

[1] On a tree of odd degree, the algorithm is more complicated. Here, we will focus on degree four, so the above description is correct.

color of each such parent is assigned as the majority vote of the children. This algorithm is repeated until it assigns a color to the root at generation $u = 0$. Parsimony has the nice property that it identifies a coloring of the tree that minimizes the number of mutations, or parent-child differences. Note, however, that this is not necessarily the maximum likelihood reconstruction. Indeed, we'll see that parsimony is a suboptimal algorithm for large p.

5 SVM

The final algorithm we consider is an SVM. We used a Gaussian kernel, with an ℓ_1 soft margin, implemented using `libsvm` [5]. We found that feeding the data of the leaves directly into the SVM leads to overfitting and relatively poor performance. Instead, inspired by the principal components discussed above, we used as features the majority vote of the descendants of diffent vertices, starting with the root and working upwards. The first feature is just the total census $N_{white} - N_{black}$ of the leaves. The second feature is the census of the first $1/(d-1)$ leaves, the third is the census of the next $1/(d-1)$ and so forth. We selected features using cross validation, and found rather robustly that the optimal number of features for the $d = 4$ tree was four. We also selected the parameters γ of the Gaussian kernel and C of the soft margin using cross validation. The results were rather insensitive to the choices of γ and C, for $\gamma \approx 0.5$ and $C \approx 30$. We trained on 20000 samples. This appears to be sufficient, judging from the learning curve shown in figure 5.

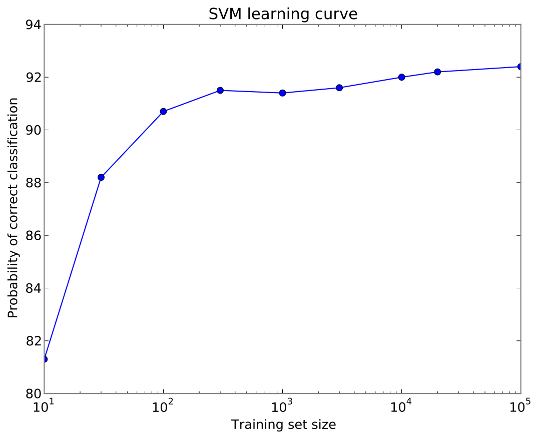

Figure 5: Learning curve for the SVM, $p = 0.11$.

6 Comparison of the methods

Finally, we tested all four of the algorithms on 20000 samples for each of the values of p. The performance is shown in the figure and table below.

As expected, the performance of all algorithms decreases monotonically as p increases. For small values of p, the parsimony algorithm does best, followed closely by the SVM. However, as p increases, parsimony is less effective. Indeed, for $p \geq 0.15$, parsimony is the worst of the four. The naive Bayes algorithm is the worst for small values of p, but it ends up more or less tied with the SVM at larger p. The SVM does best overall, coming close to the accuracy of parsimony for small p, but tracking the naive Bayes performance for larger p. NSNB has performance very similar to the SVM, but it is slightly worse for all p.

We conclude that the SVM emerges, once again, as having very competitive performance across the board.

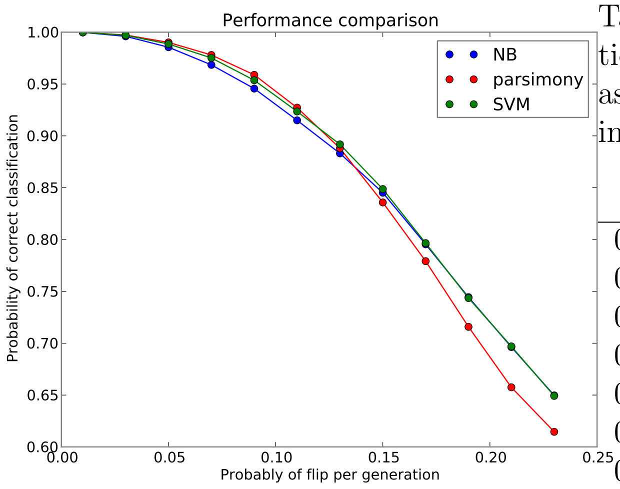

Figure 6: Performance of different reconstruction algorithms for various values of p. The degree is fixed at four, and simulation is run for six generations. The NSNB curve is omitted for clarity. See the table for more details.

Table 1: Probability of correct reconstruction for the five reconstruction algorithms, as a function of p, the probability of changing color in a single generation

p	NB	parsimony	NSNB	SVM
0.01	0.9998	0.9998	0.9998	0.9998
0.03	0.9960	0.9972	0.9969	0.9970
0.05	0.9855	0.9901	0.9886	0.9886
0.07	0.9684	0.9780	0.9751	0.9751
0.09	0.9456	0.9588	0.9511	0.9536
0.11	0.9150	0.9272	0.9198	0.9235
0.13	0.8831	0.8880	0.8836	0.8918
0.15	0.8451	0.8358	0.8411	0.8487
0.17	0.7954	0.7791	0.7875	0.7965
0.19	0.7444	0.7158	0.7408	0.7435
0.21	0.6963	0.6575	0.6927	0.6969
0.23	0.6497	0.6145	0.6462	0.6492

References

[1] T. Moore and J. L. Snell. A branching process showing a phase transition. *Journal of Applied Probability*, 16(2):pp. 252–260, 1979.

[2] P. Bleher, J. Ruiz, and V. Zagrebnov. On the purity of the limiting gibbs state for the ising model on the bethe lattice. *Journal of Statistical Physics*, 79:473–482, 1995. 10.1007/BF02179399.

[3] H. Kesten and B.P. Stigum. Additional limit theorems for indecomposable multidimensional galton-watson processes. *Ann. math. Statist.*, 37:1463–1481, 1966.

[4] E. Mossel. Reconstruction on trees: beating the second eigenvalue. *The Annals of Applied Probability*, 11(1):285–300, 2001.

[5] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.