Efficient Calculation of Interaction Features on Sparse Matrices

Andrew Nystrom

Abstract

FILL THIS IN

1 Introduction

Introduction Interaction features are a way of capturing correlations between features in a machine learning setting. They consist of products of combinations of features. This work describes a method for efficiently calculating second degree interaction features on a sparse matrix, and could be generalized to higher orders.

Consider the following matrix:

$$\begin{pmatrix} 3 & 0 & 0 & 3 \\ 0 & 4 & 2 & 0 \\ 1 & 2 & 0 & 3 \end{pmatrix}$$

If each row is an instance vector, then the interaction feature matrix is

$$\begin{pmatrix} 0 & 0 & 9 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8 & 0 & 0 \\ 2 & 0 & 3 & 0 & 6 & 0 \end{pmatrix}$$

Note that there are $\binom{D}{2} = \frac{D^2-D}{2}$ columns in the interaction matrix, which is D choose 2, since we generate products of all combinations of 2 features

in the original matrix. Which column corresponds with which product pair is not important so long as its consistent. In this example, each column in the interaction matrix corresponded to the following product pairs: $(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)$.

Notice that the interaction matrix contains many zero entries. This is of course because the original matrix contained zero entries, so the interaction features, which are products of pairs of features, contain many zeros. This means that the only products that need to actually be calculated or those for which both features in the combination are nonzero. If the original matrix is sparse and represented in a sparse matrix format (e.g. compressed sparse row), a list of nonzero column indices are stored for each row and are easily retrievable in $O(1)$ time. Interaction features can be generated from this list via the following method.

2 Approach

Let the list of nonzero columns for a given row be N_{zc}. The nonzero second degree interaction features are simply the products of all combinations of two elements whose columns are in N_{zc}. However, to do this efficiently, a consistent mapping from the column index pairs of N_{zc} into the columns of the interaction matrix is needed. The mapping is from the space (a, b) where a,b are in $1, 2, ..., D$ onto the space $1, 2, ..., \frac{D^2-D}{2}$. This is isomorphic to mapping the coordinates of the upper triangle of a matrix onto a flat list. The following is a proof by construction for such a mapping.

INSERT JOHN'S PROOF HERE

With this mapping, an algorithm for generating second degree interaction features on a matrix A can be formulated as follows:

SPARSE INTERACTION(A)

$\text{map}(b, a) = \frac{2Da - a^2 + 2b - 3a - 2}{2}$
B = Compressed Sparse Row Matrix of size $N \times \frac{D^2 - D}{2}$
for *row* in A **repeat**
  N_{zc} = nonzero columns of *row*
  for each combination a, b of elements of N_{zc} **repeat**
    $k = \text{map}(b, a)$
    i = index of *row*
    $B[i, k] = row[a] \cdot row[b]$

3 Complexity Analysis

Assume that A is a matrix with sparsity $0 < d < 1$, N rows, and D columns. Finding interaction features with the proposed algorithm has time and space complexity $\text{O}(dND^2)$, whereas a naive approach of using non-sparse matrices and multiplying all column combinations has time and space complexity $\text{O}(ND^2)$. The algorithm is therefore an improvement by a factor of the density factor of A.

This can represent a large gain in speed and time. For example, the 20 Newsgroups dataset has density d of 0.12 when its unigrams are represented in a vector space model. This means the proposed approach would take less than $\frac{1}{8}$ time and memory.

The real benefit of this method is revealed when the average complexity is analysed. The number of interactions calculated for a given row are $\binom{|N_{zc}|}{2}$. If the matrix has density d, then on average, $N_{zc} = Dd$, so the number of interaction features calculated in total is

$$N\binom{dD}{2} = \frac{N(Dd)!}{2!(Dd-2)!}$$

$$= N\frac{(D^2d^2 - Dd)}{2}$$

This means that the average complexity decreases quadratically with the

density.

4 Future Work

The approach for generating second degree interaction features required a mapping from combinations of two to the space $1, 2, ..., \frac{D^2-D}{2}$, which is isomorphic to a mapping from the indices of an upper triangular matrix to the indices of a flat list of the same size. To generate third degree interaction features, a mapping from combinations of three (a, b, c) to the space $1, 2, ...\frac{D^3-3D^2+2D}{6}$ (which is $\binom{D}{3}$), or the upper 3-simplex of a tensor to a flat list of the same size $\frac{D^3-3D^2+2D}{6}$ would be required. In general, for interaction features of degree k, the upper k-simplex of a k-dimensional tensor must be mapped to the space $1, 2, ...\frac{D!}{k!(D-k)!}$. A similar approach for finding these mappings could be taken as the one used here for $k = 2$.

Motivation for deriving mapping functions for higher orders of interaction features is that the average complexity of generating degree k interaction features is $N\binom{Dd}{k}$, which decreases polynomially with respect to k compared to generating the features naively.