CS6190: Probabilistic Modeling Project

Gopal Menon

02 May, 2018

Project Description

I implemented a dimensionality reduction algorithm called t-SNE (t-Distributed Stochastic Neighbor Embedding) [1] that is used for visualizing high dimensionality data sets. It essentially works by mapping points in high dimensions to points in low dimensions by minimizing the KL Divergence between two probability distributions. The first distribution is for the probability of picking a point as a neighbor of another point in high dimensional space and the second one is for the corresponding distribution in low dimensional space. The algorithm works by starting with a random distribution of points in low dimensions and does a gradient descent on the KL Divergence gradient using a learning rate and a momentum term. The latter used to speed up optimization and hopefully avoid local minima.

t-SNE short description

First high-dimensional distances between points are converted to conditional probabilities that represent similarities [1]. The similarity of datapoint x_j to datapoint x_i is the conditional probability $p_{j|i}$ that x_i would pick x_j as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at x_i. The conditional probability $p_{j|i}$ is given by

$$p_{j|i} = \frac{exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} exp\left(\frac{-\|x_i - x_k\|^2}{2\sigma_i^2}\right)}$$

where σ_i is the variance of the Gaussian that is centered on datapoint x_i. $p_{i|i}$ is set to zero.

A Student t-distribution with one degree of freedom is used as the low-dimensional distribution. The joint probability q_{ij} in the low-dimensional representation is defined as

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|y_k - y_l\|^2)^{-1}}$$

$q_{i|i}$ is set to zero.

For each point high-dimensional point x_i we need to select the variance σ_i of the Gaussian that is centered over it. This is done by doing a binary search for the value of σ_i that produces a P_i with a fixed perplexity that is specified by the user. The perplexity is defined as

$$Perp(P_i) = 2^{H(P_i)}$$

where $H(P_i)$ is the entropy of P_i defined as

$$H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}$$

The perplexity can be interpreted as a smooth measure of the effective number of neighbors.

The gradient of the KL Divergence between the high dimensional symmetrized conditional probabilities $p_{ij} = \frac{p_{j|i}+p_{i|j}}{2n}$ and the low dimensional joint probability q_{ij} is given by

$$\frac{\partial C}{\partial y_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}$$

For a detailed description, please refer to the t-SNE paper [1].

t-SNE Algorithm

Algorithm 1 t-Distributed Stochastic Neighbor Embedding

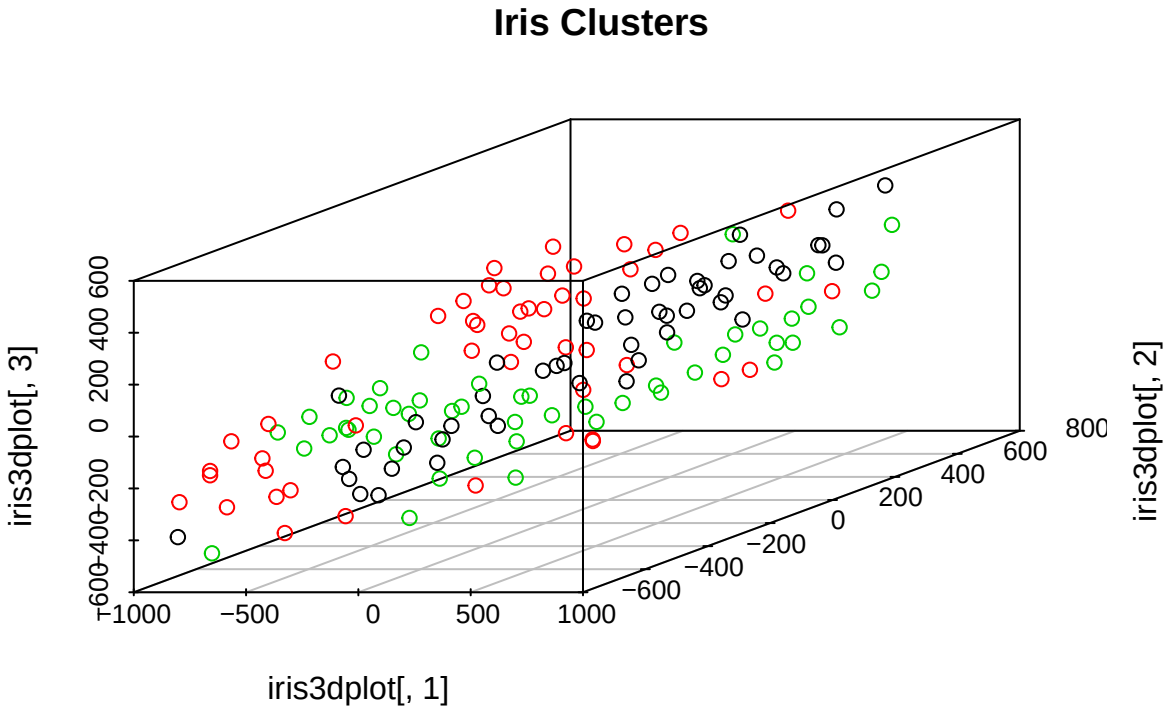
procedure T-SNE
  Data: data set $X = \{x_1, x_2, \ldots, x_n\}$,
  cost function parameters: perplexity *Perp*
  optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$
  Result: low-dimensional data representation $\gamma^{(T)} = \{y_1, y_2, \ldots, y_n\}$,
  begin
    compute pairwise affinities $p_{j|i}$ with perplexity *Perp*
    set $p_{ij} = \frac{p_{j|i}+p_{i|j}}{2n}$
    sample initial solution $\gamma^{(0)} = \{y_1, y_2, \ldots, y_n\} \sim \mathcal{N}(0, 10^{-4}I)$
    for $t = 1$ to T **do**
      compute low-dimensional affinities q_{ij}
      compute gradient $\frac{\partial C}{\partial \gamma}$
      set $\gamma^{(t)} = \gamma^{t-1} + \eta\frac{\partial C}{\partial \gamma} + \alpha(t)(\gamma^{(t-1)} - \gamma^{(t-2)})$
  end

Implementation Details and Issues

The algorithm shown above was implemented and needed tuning of the hyperparameters for perplexity, learning rate and momentum for getting good results. In addition to that, the equation for the next low dimensional estimate $\gamma^{(t)} = \gamma^{t-1} + \eta\frac{\partial C}{\partial \gamma} + \alpha(t)(\gamma^{(t-1)} - \gamma^{(t-2)})$, was changed so that the second term is subtracted. I am not sure if that was a typo in the paper or not, but it gave me better results and it also made sense to take a step in the direction of the negative gradient.

Experimental results with Iris dataset

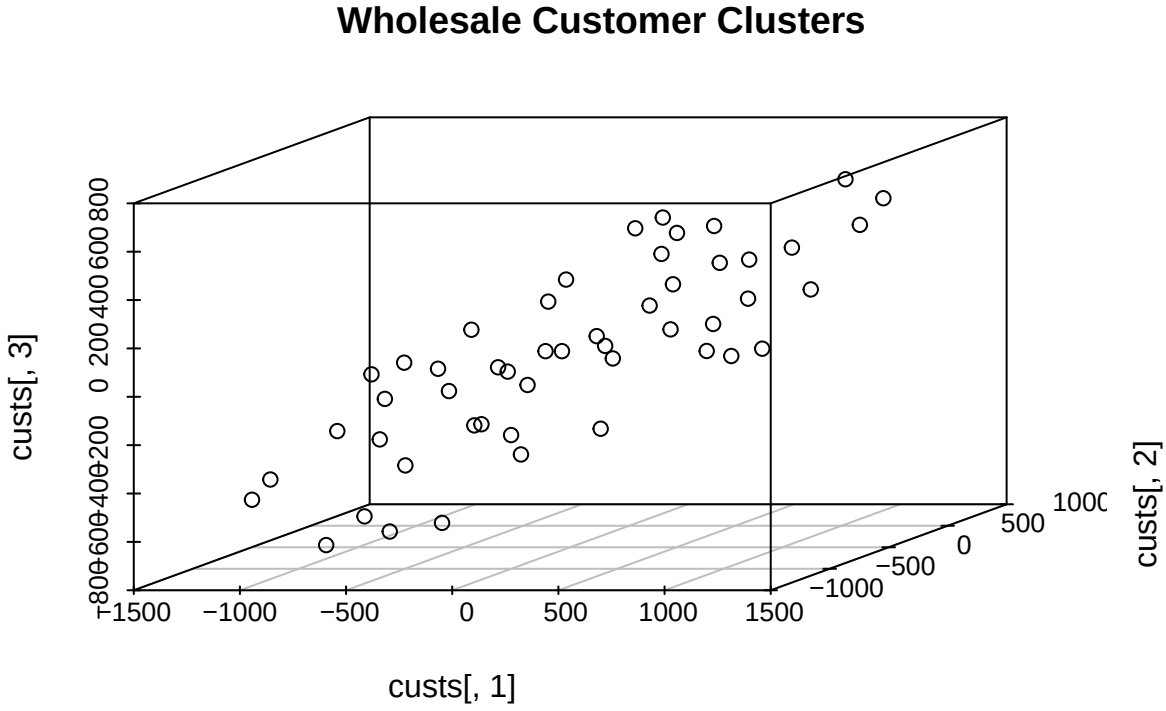
```
## Warning: package 'scatterplot3d' was built under R version 3.4.4
```



The iris dataset was reduced from 4 to 3 dimensions usign t-SNE using hyper-parameters that were optimized using trial and error. The results do not show perfect clusters, but do show clusters of three different colors corresponding to the three types of iris flowers. t-SNE is supposed to be able to achieve clusters that are widely separated, however I was not able to achieve that. I needed to put in the colors to be able to separate out the flower types.

Experimental results with Wholesale Customer Dataset

I was planning to use a wholesale customer dataset published by UC Irvine at https://archive.ics.uci.edu/ml/datasets/Wholesale+customers. I could not get good clustering results using this dataset. The problem was similar to the one with the Iris dataset. There were no well defined clusters. And since the true clusters were not available, I was not able to validate my results. Also, I ran the clustering with only the first 50 rows as t-SNE is very compute intensive.



Issues

I was not able to implement a fully working t-SNE algorithm. There seems to be a problem somewhere that I cannot figure out. I think the two weeks I had were not enough and I was over-optimistic.

References

[1] Maaten, Laurens van der, and Geoffrey Hinton. "Visualizing data using t-SNE." *Journal of machine learning research*, 9.Nov (2008): 2579-2605.