Comparison of Dimensionality Reduction Techniques

Jonathan Tan
Harris School of Public Policy
University of Chicago
Chicago, IL 60637
jonathantan@uchicago.edu

Satej Soman
Harris School of Public Policy
University of Chicago
Chicago, IL 60637
satej@uchicago.edu

Abstract

We compare two dimensionality-reduction techniques: principle components analysis and latent space representation via autoencoder neural networks. Both techniques are applied to a feature set and the reduced feature-set is used as the independent variables in a linear regression to recover a known score. We find that XXXXXX. We also discuss considerations such as explainability and training-time.

1 Background

Dimensionality reduction is an important technique to render high-dimensional datasets more tractable for analysis. Often, in a dataset containing thousands of columns, a small number of those columns are relevant for an analytical task. Robust techniques for identifying the most salient feature techniques allow for more compact representations of data, reducing storage costs and analysis time. [insert reference XXXX]

2 Techniques studied

2.1 Principal components analysis

Principle components analysis is

2.2 Autoencoders

In contrast, autoencoders seek to map inputs to a *non-linear* input manifold.

We test autoencoders with both logistic activation as well as ReLU activation.

3 Dataset

3.1 Description

We use a dataset of wine reviews from Kaggle [insert reference XXXX] consisting mostly of categorical and text data, each with an associated wine review score on a scale of 80 - 100. The features include a free-text description of the wine, as well as pre-parsed information about the wine, including type, grape variety, region of origin, and winery. The key continuous, numeric column in the feature set is the price of each wine bottle. The label we seek to recover is the wine review score.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

3.2 Preprocessing

Much of the Kaggle dataset is text or categorical data. Specifically, the description feature contains paragraph text of descriptors for each wine. To obtain a numeric representation of this text data, we used a simple boolean bag-of-words model. We first preprocessed the text data by converting all words to lowercase, removing non-alphanumeric characters and common stopwords, then tokenizing the space-delimited string into separate words. Lastly, we computed a bit vector for each row that represented a one-hot encoding for each word in the description, over all unique words in all rows of the description feature. [insert reference XXXX]

The bag-of-words model gives a particularly sparse matrix, and we faced initial difficulties in computing the representation with the full 130,000-row dataset (which had roughly 45,000 unique tokens). Specifically, the resulting matrix from the description column alone was over 100GB and, when the process ran in a reasonable amount of time, we had difficulty storing the data on our local machines. As an interim solution, we proceeded with a reduced dataset, taking 10,000 rows from the wine dataset instead.

In the future, other numeric representations of text data could be interesting to explore, e.g. low-dimensional word embeddings produced by neural network-based algorithms such as `word2vec` or `GloVe`. [insert reference XXXX]

4 Methodology

For each technique, we apply the dimensionality reduction to obtain a reduced set of feature vectors (X_j). We then regress wine price (P_i) on these reduced features and look at the mean-squared error.

5 Results

6 Conclusions

7 Future work

There are a number of ways this work can be extended. As dicussed above, we could use alternate encoding schemes for the free-text data. More interesting lines of inquiry involve either different neural-network architectures, different loss functions, or alternative reduction techniques.

Our neural networks were not symmetric; after a number of encoding layers, we simply had one decoding layer and penalized the neural network for its inability to recover the original input vector. Autoencoders with the same number of input and output layers likely allow for more precise decoding and thus faster convergence in training.

Additionally, we could have used regression mean-squared error as the loss function when training neural networks, but there is not a clear analog of this when doing principal-components analysis, so we opted to simply force the decoded output to be as close as possible to the unencoded input.

Finally, we hoped to compare the techniques discussed to another technique called t-Distributed Stochastic Neighbor Embedding (t-SNE) [insert reference XXXX].

References

[1] Alexander, J.A. & Mozer, M.C. (1995) Template-based algorithms for connectionist rule extraction. In G. Tesauro, D.S. Touretzky and T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609–616. Cambridge, MA: MIT Press.

[2] Bower, J.M. & Beeman, D. (1995) *The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System.* New York: TELOS/Springer–Verlag.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.