# Text Message Classification with XGBoost

Kamden Baer, Charlotte Imbert, Quinn Link

December 15, 2023

## Introduction

When faced with the opportunity to do anything with data, we turned inward to think about the ways that we create data ourselves every day. One of these ways is through the texts that we send—the thousands of messages, characters, emojis, and emotions we use and communicate with each other. The people we communicate with matter too, whether it be our friends, family, or partners. We wanted to investigate the ways in which we leverage all of these different texting options to communicate across these different groups. In sociological and psychological studies, the Communication Accommodation Theory proposes that we adjust our communication styles to either converge or diverge from others in social settings (Giles & Ogay, 2007). We aim to test how well this theory can be observed within texting data and if there are important indicators that can explain how communication styles converge to or diverge from others via text.

The data we decided to work with comes from the texting data of Quinn, one of our project members. He downloaded his texts from 2016 to the beginning of December 2023, preprocessed them for some experimental analysis and data visualization, and eventual classification. We attempted to build a learner capable of differentiating between texts Quinn sent to his girlfriend, friends, or family, using various features of the data to inform our predictions. One feature we hoped would have considerable predictive power was some sort of sentiment score we could assign to each text. We conducted sentiment analysis on the dataset and then used the assigned scores as a feature to predict each text's recipient. Finally, we used an XGBoost model to classify texts by recipient given our predictor variables.

## Data Collection and Preparation

We collected text message data from Quinn's iPhone on December 7th 2023. The data encompassed text messages from April 2016 to December 2023. The raw data from text messages had to be cleaned in order to remove observations that did not contribute towards determining written communication style; such observations included notifications for reactions to texts, as well as minigames, links and photos. After cleaning, the data consisted of 286,898 observations: 266,429 of Class 0 (girlfriend), 4682 of Class 1 (friends), 15787 of Class 2 (family). The initial predictor variables were as follows: time between messages (seconds, determined by using the 'lubridate' package to calculate the difference between the timestamps of two successive messages), the number of words, the number of characters, and the number of emojis. Data about each text message also included whether it was an incoming or an outgoing message, as well as the text message itself. As the main purpose of this analysis was to classify texts by who they were sent to (Class 0, 1 or 2), during the classification section of the data analysis the data set was filtered to contain only outgoing text messages.

## Sentiment Analysis

Sentiment analysis was carried out on the text contained in each message, using multiple R packages including tm, tidytext and SnowballC. Sentiment analysis is a text mining method that can be used to assign a sentiment score to each text message, with higher positive values being associated with positive emotions, and lower negative values denoting negative emotions. In order to maximize the accuracy of the sentiment analysis, each text message was cleaned to convert each word to lowercase, remove punctuation, remove emojis, and remove stopwords, which are commonly used words that do not have sentiment attached to them, such as 'the'. Negations were also detected and handled, as words indicating negation significantly alter the meaning of a sentence and thus its sentiment. For this particular project, the AFINN lexicon was used to calculate sentiment scores for each text message. Average sentiment score was added to the dataset as a feature. Since texts to certain people, such as a partner, tend to express higher sentiment than texts to others, it was hypothesized that this feature might have predictive power in the classification task.

## Imbalanced Data and SMOTE

The dataset is massively imbalanced with over 92% of the data belonging to Class 0. An imbalanced data set poses several challenges for classification problems. Models designed to optimize overall accuracy may predict the majority class for most, or all, of the observations. Therefore a learner might fail to learn anything about the minority class and the generalizability of the model will be negatively impacted, especially if the class distribution is different from the training set.

Our chosen model, XGBoost, seeks to maximize information gain at each split. Gain is a measure of how much more "pure" the child nodes are compared to the parent node. However, seeking to maximize gain may cause the learner to overlook minority classes, as the gain of separating these classes is negligible compared to the gain the learner can achieve by classifying the majority class well. What is more, XGBoost seeks to create leafs that are pure, so minority class leafs could be rare and based on very few examples. These minority class leafs may not be reliable as they were not trained on enough data.

To address this issue we utilized Synthetic Minority Oversampling Technique (SMOTE). The algorithm works by identifying minority classes that need to be upsampled to counter the overall imbalance. SMOTE then interpolates using existing data points to create synthetic observations of the minority classes. It does this by identifying a minority class data point's K nearest neighbors of the same class and then calculating the difference between the feature vector of each point and its neighbors. Finally, this weighted difference is added to the original observation to create a new data point (Chawla et al, 2002). We implemented SMOTE to increase the number of observations in class 1 from 3718 to 26,026 and the number of observations in Class 2 from 12653 to 88,571.

## Exploratory Data Analysis

Although the main purpose of adding sentiment score as a feature was to see its impact on classification performance, sentiment analysis also was valuable in revealing trends in messages over time. For this part of our analysis, both outgoing and incoming messages were considered for the 'girlfriend' dataset. A separate mean sentiment score was assigned for outgoing texts and incoming texts for each year, from 2016 through 2023. This was done to show trends in sentiment over time as well as trends in sentiment between the individuals in the relationship.

We explore the sentiment score over time from the largest segment of the data, texts to and from girlfriend. We can see that initially in 2016, there is a large imbalance between the average sentiment score of texts to versus from girlfriend. Later, the sentiment scores are more consistent with one another, with texts to girlfriend having higher sentiment scores on average than texts from girlfriend. The average sentiment scores for texts Quinn sent to his girlfriend were between 1.4 and 1.8 for every year after 2016, while the average sentiment scores for the texts she sent him were between 1.2 and 1.6. The change in sentiment scores of texts sent to his girlfriend indicates that texting style may not be consistent in this category across time. Therefore we removed observations from 2016 and 2017 before training our learner.

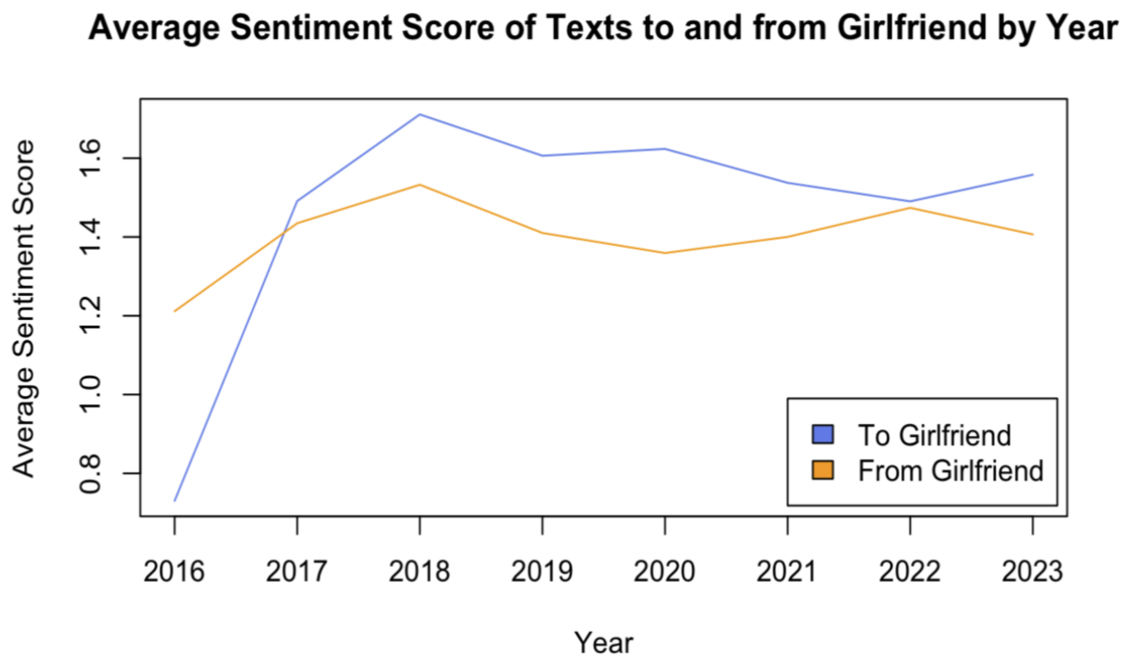

Figure 1: Trends in the average text sentiment scores for messages sent to and from girlfriend between 2016 and 2023.

## XGBoost Classification

We used XGBoost as the base model for our learner. To tune our learner we used a nested cross-validation process with k =3 folds using the mlexperiments package. The code for our tuning procedure and implementation can be found in the appendix. We tuned our learner using Balanced Accuracy (BACC) as our performance metric. Balanced accuracy is useful when dealing with imbalanced datasets because it weighs the performance of the model on each class equally (see appendix).

### Results

Even with a balanced dataset, XGBoost struggled to classify the text messages accurately. We tested our model using the best combination of parameters from all three of our folds, and these models consistently performed poorly with an average BACC of approximately 0.375.

Table 1: Results of XGBoost on Balanced and Unbalanced Data (BACC Score)

| **XGBoost on Unbalanced Data** | |
|---|---|
| **Model** | **BACC Score** |
| Fold 1 | 0.3612946 |
| Fold 2 | 0.3609054 |
| Fold 3 | 0.3632493 |
| **XGBoost on Balanced Data** | |
| **Model** | **BACC Score** |
| Fold 1 | 0.3737194 |
| Fold 2 | 0.3768982 |
| Fold 3 | 0.3765224 |

### Feature Importance

We quantified feature importance using a number of metrics. For each feature we calculated the average gain of splits which used that feature (see appendix). A higher gain value indicates a greater contribution to the model's prediction accuracy. Coverage is a measure of the number of observations affected by the split. A higher value means the feature affects more observations. Frequency shows how often a feature is used to split the data. The table also contains a measure of overall "Importance" which is a normalized score derived from the other metrics.

We see that time between messages is the most important feature in our learner, indicating that the content of the message is less important in differentiating between classes than the pace at which the communication takes place. The number of words and characters as well as the sentiment score were relatively important while punctuation and number of emojis did not significantly contribute to the decision boundary created by our learner.

Table 2: Feature Importance from XGBoost Classifier

| **Feature** | **Gain** | **Cover** | **Frequency** | **Importance** |
|---|---|---|---|---|
| TimeBetweenMsg | 0.35661298 | 0.38923356 | 0.35492155 | 1.00000000 |
| NumWords | 0.20924036 | 0.16940112 | 0.18975054 | 0.58674353 |
| NumChar | 0.18142499 | 0.26350870 | 0.21224179 | 0.50874476 |
| avg_score | 0.16344841 | 0.10491590 | 0.14050683 | 0.45833556 |
| NumCasPun | 0.05443179 | 0.02854845 | 0.05762501 | 0.15263547 |
| NumEmojis | 0.02018414 | 0.02237670 | 0.02655492 | 0.05659955 |
| NumFormPun | 0.01465733 | 0.02201557 | 0.01839937 | 0.04110152 |

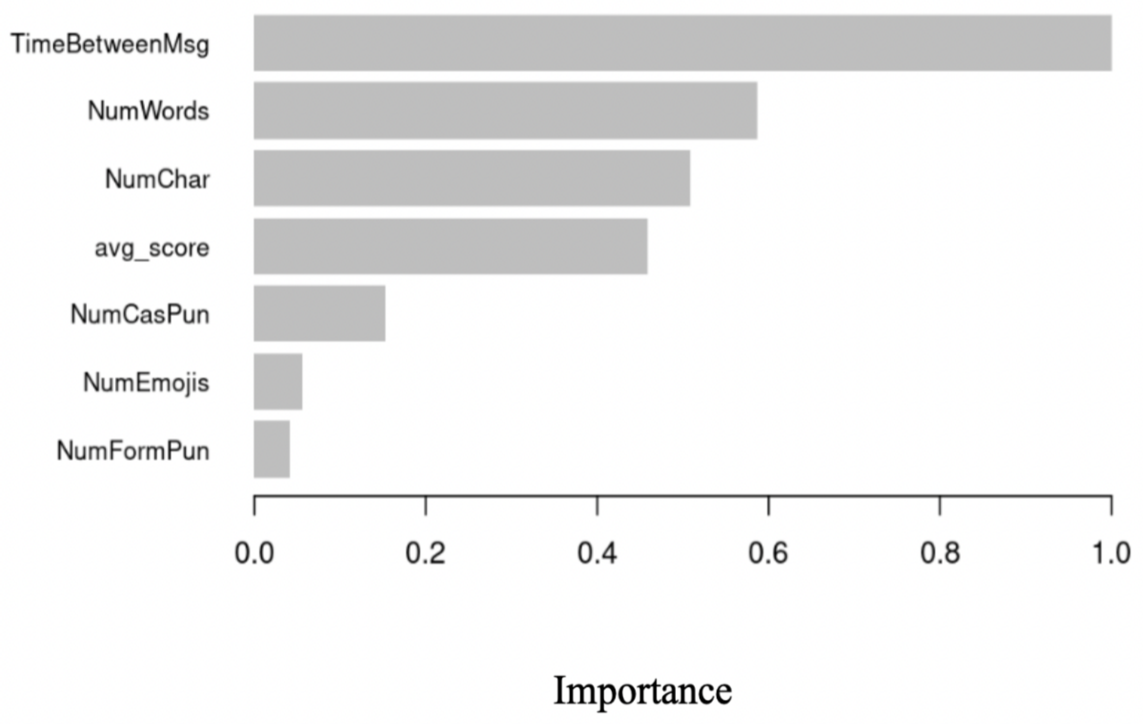

Figure 2: Feature Importance from the XGBoost Classifier

## Data Visualization

### PCA

We performed PCA in an attempt to reduce the dimensionality of our data. However, it appears that our data lives mostly in 5 dimensions. Therefore the 2D representation of our data using principal components 1 and 2 is unlikely to be a faithful representation of our data.

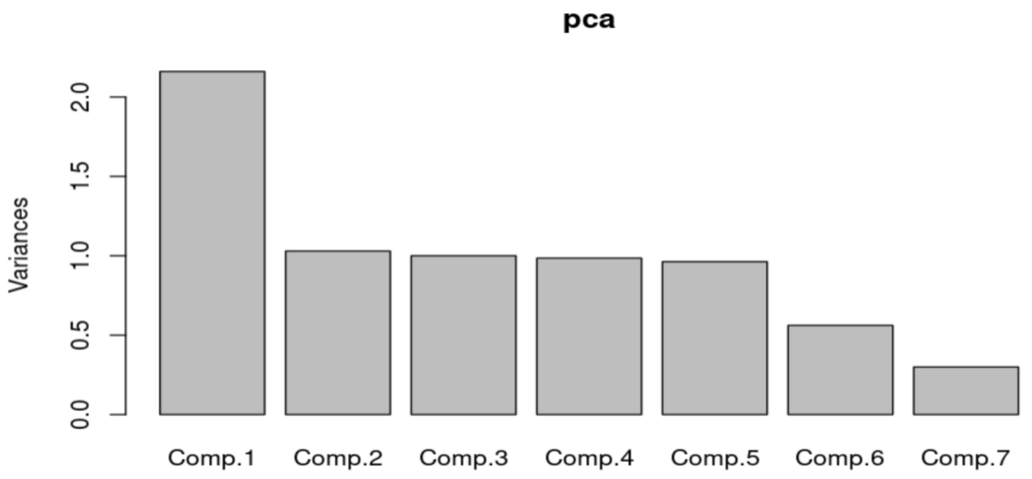

Figure 3: Top 7 principal components and their percentage of variance explained, obtained via PCA.

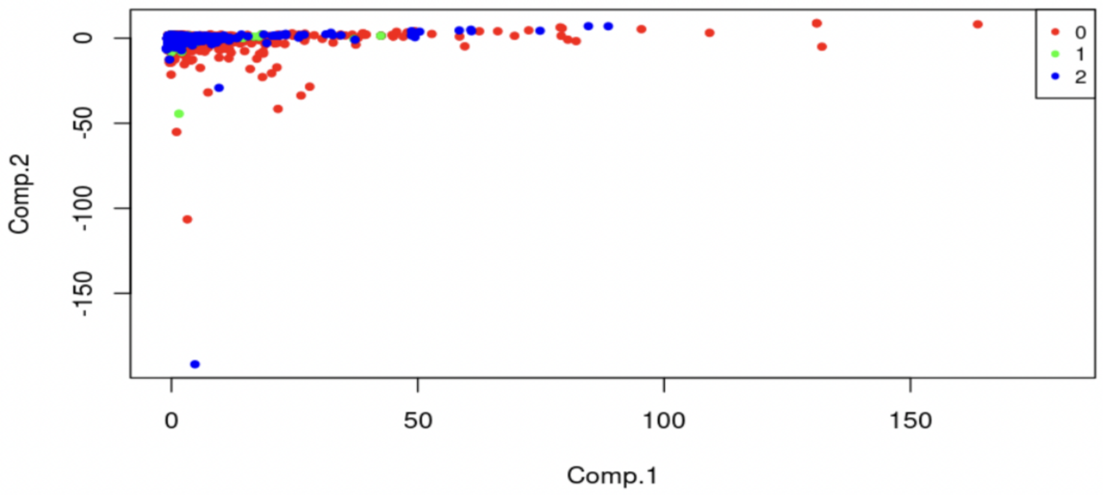

Figure 4: Relationship between the top 2 principal components, with 1, 2, and 3 referring to text message labels.

## Mapper

Another data visualization we used was 'Mapper', first described by Singh, Mémoli and Carlsson in 2007. The aim of Mapper is to extract 'simple descriptions of high dimensional data sets in the form of simplicial complexes' (Singh et al., 2007). Clustering is a crucial part of this algorithm and there is freedom in the clustering technique used. In our analysis, Mapper was implemented using the TDAmapper package in R. Mapper is valuable in that it can build 'useful combinatorial representations of geometric information about high dimensional point cloud data' while only requiring 'knowledge of the distances between points and a choice of combination of filters' (Signh et al., 2007).

We performed mapper using the first principle component as the filter value. With this choice of filter value, Mapper was able to uncover some of the underlying geometry of the data. However, the map did not successfully create meaningful clusters of the classes in the data. The nodes that are majority Class 2 only contain one datapoint and there are no nodes with a majority of Class 1. When we scale node size by the number of data points contained within, we see that the majority of the data lives in the first node, indicating that the data exists mostly in the same region of the data space.

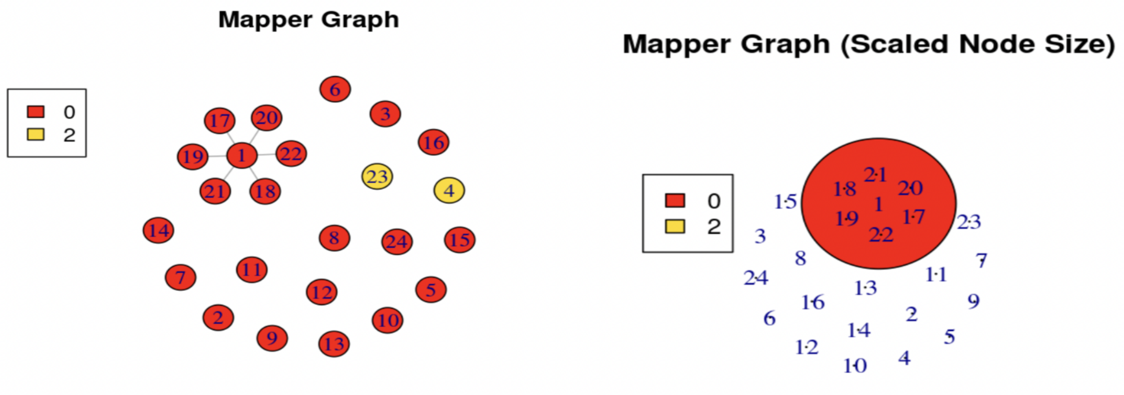

Figure 5: Clustering of the data into nodes using TDAmapper in R.

## t-SNE

Another data visualization technique employed in this analysis is t-Distributed Stochastic Neighbor Embedding (t-SNE). Like many data visualization techniques, t-SNE performs dimensionality reduction, which aims to map data to a lower-dimensional map while still maintaining the important structure of the original higher-dimensional data. In t-SNE, a high-dimensional data set is first converted into 'a matrix of pairwise similarities'; t-SNE then allows the similarity data produced to be visualized (Hinton & van der Maaten, 2008). This method is advantageous in that it not only captures 'much of the local structure of the high-dimensional data very well' but it also informs us about 'global structure such as the presence of clusters at several scales' (Hinton & van der Maaten, 2008). t-SNE was used in our analysis because of its benefits over other data visualization techniques: it can be optimized much more easily than Stochastic Neighbor Embedding (SNE), and by 'reducing the tendency to crowd points together in the center of the map' allows for more effective and clearer visualizations (Hinton & van der Maaten, 2008). It is also capable of displaying the structure of big data sets, such as the one used in this study, by using 'random walks' on subsets of the data.

We used t-SNE as an alternative to PCA and Mapper to visualize our data. The results of t-SNE reveal some of the underlying structure of the data, however we can see that all three classes tend to live in the same regions of the dataspace. This supports our findings with XGBoost and Mapper that these text messages are not separable by class.

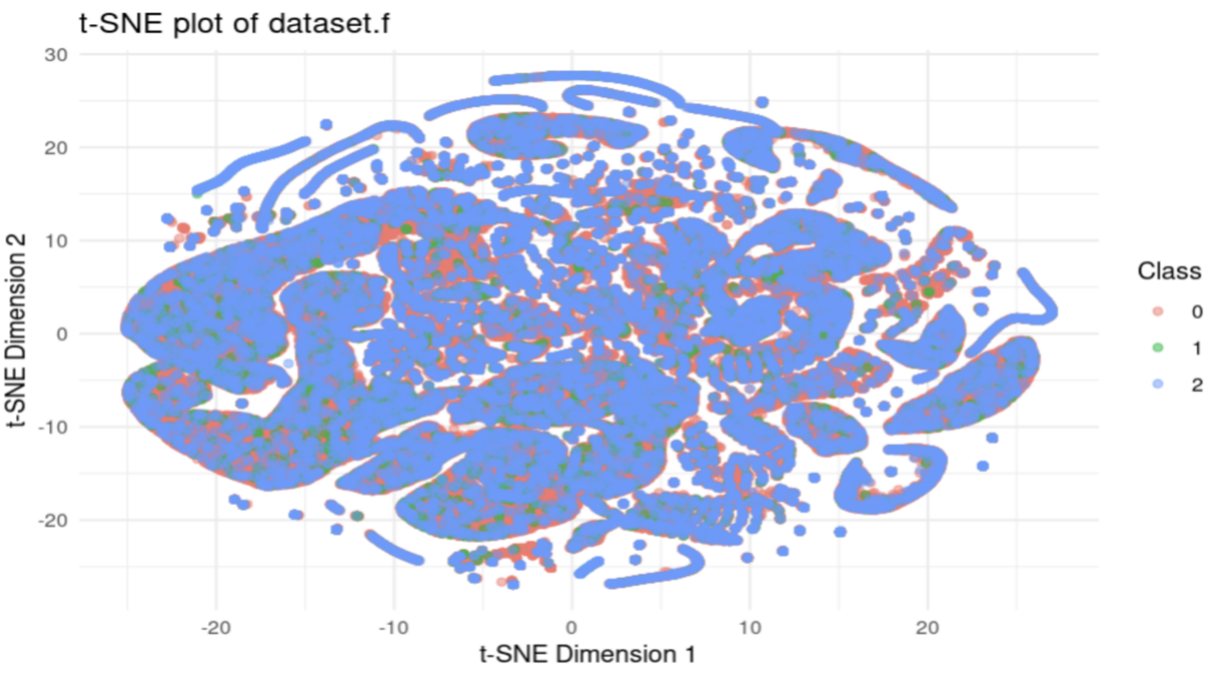

Figure 6: t-SNE output, showing that all three classes of text messages tend to live in the same regions of the dataspace.

## Conclusion

We harness a variety of algorithms—PCA, Mapper, and t-SNE—to attempt to visualize the separation of Quinn's text data to his girlfriend, friends, and family. None of these techniques are able to successfully separate the data into meaningful categories, indicating that the classes in data cannot be well separated. Given these results, we conclude that Quinn texts his girlfriend, friends, and family in similar communication styles, or at least among the communication styles captured in the data we generated.

Aside from trying to solve the classification problem of sorting text messages by who they were sent to, we also examined the trend in sentiment analysis over time between the messages exchanged between Quinn and his girlfriend. With the exception of 2016, outgoing messages Quinn sent to his girlfriend showed consistently higher sentiment on average than the ones she sent to him. This analysis also revealed that sentiment for both outgoing and incoming texts did not systematically increase or decrease over the years.

## Works Cited

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16, 321-357.

Giles, H., & Ogay, T. (2007). Communication Accommodation Theory. In B. B. Whaley & W. Samter (Eds.), Explaining communication: Contemporary theories and exemplars (pp. 293-310). Lawrence Erlbaum Associates.

Singh, G., Mémoli, F., Carlsson, G. (2007). Topological Methods for the Analysis of High Dimensional Data Sets and 3D Object Recognition. Eurographics Symposium on Point-Based Graphics.

Van der Maaten, L., Hinton, G. (2008). Visualizing Data using t-SNE. Journal of Machine Learning Research, 9, 2579-2605.