Napa (Tan) Vananupong

CS 4641 Machine Learning

April 7 2019

Dr. Brian Hrolenok

Assignment 3: Unsupervised Learning

In this assignment, I will be examining the many interesting problems and techniques you can explore with unsupervised learning algorithms, specifically Clustering algorithms and dimensionality reduction algorithms. For Clustering algorithms, I will be exploring K-means clustering and expectation maximization (EM). For Dimensionality reduction algorithms, I will be exploring Principal Component Analysis (PCA), Independent Component Analysis (ICA), Randomized Projections (RP), and a chosen feature selection algorithm called Information Gain that I selected from WEKA. For the Principal Component Analysis, I will examine the effects of using the technique before clustering by using PCA as a feature transformation method for the neural network, vs just using plain clustering as features for the neural networks. The goal of this assignment is for me to develop a better understanding of unsupervised learning algorithms, effects of clustering, and feature selection, especially how and why some algorithms perform better than others and in which situations each algorithm will have an advantage in.

Introduction on Datasets

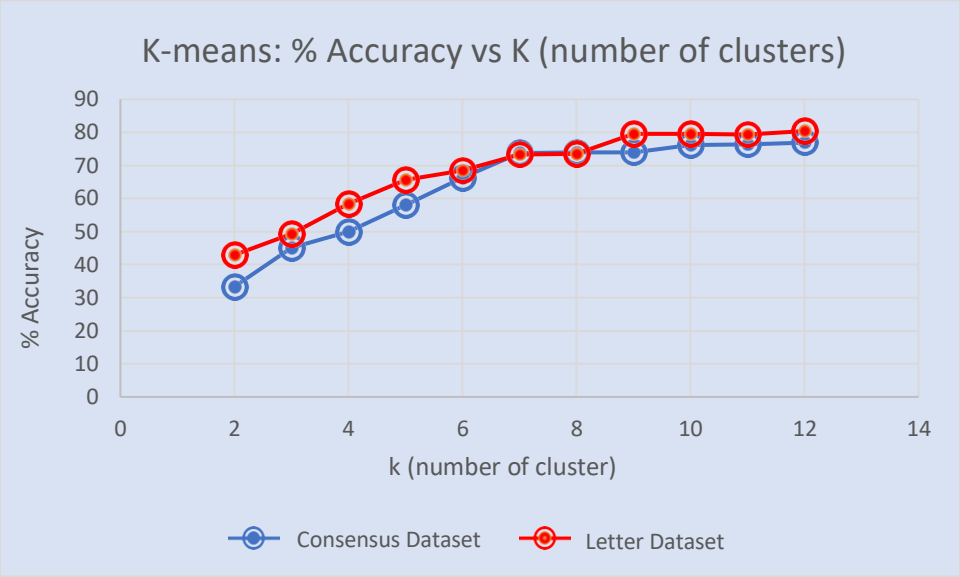
For this assignment, I am reusing the same datasets from my first assignment, which I found on Kaggle. The aforementioned datasets are a dataset of the National Consensus data and a dataset of Letter data (data about letters of the English alphabet). The first dataset was interesting to me because I thought it would be cool to predict the education level of each individual in the

United States based on various attributes such as their gender, age, race, marital status, etc. Especially as a woman, I really wanted to see if gender actually made a difference in educational level if other attributes were similar. Thus, by predicting education level, I wanted to see if for any reason the algorithms would predict higher education for males than females with similar other attributes. Furthermore, this is a useful problem because it allows us to consider if educational level actually matters in the long run, which, if looked at by a human, would basically indicate if you should actually pursue higher education in order to achieve a better income, for example. I chose the Letter dataset mainly because it contained a lot of good attributes to work with and was rather large, and in my opinion, more 'unpredictable' since letters are so random in shape. I also chose this one because I wanted to test the unsupervised learning algorithms on something completely different than the first dataset. Whereas consensus data is probably a very popular choice to use for machine learning, I felt that this would be a more unique approach: using the unique attributes of each of the English alphabet's characters to predict which character they belonged to. This dataset is also had a vast collection of data – over 20,000 entries and sixteen features. Throughout this paper, I will refer to the first dataset as the National Consensus Dataset, and the second as the Letter dataset. For all experiments, I am using WEKA, the machine learning framework I used earlier for assignment one, which is available for download here: https://sourceforge.net/projects/weka/.

K means

The first algorithm I applied was K-means. K means clustering is a simple yet powerful clustering algorithm. The algorithm partitions data into clusters where the organization is more effective for better results. It starts off by picking K centers at random, which usually are points of data. Each 'center' then selects all the points of data that are closer to it than the other centers,

thus forming K ‘clusters’. This process continues until the center point no longer moves from recalculation of the average location of cluster points. For the clustering algorithms, I used the percentage of incorrectly clustered data as a metric. One limitation in K-means is that we can only optimize one hyperparameter, the K. The user has to manually select the K, which means we do not know beforehand the optimal k value, but have to use a guess-and-check approach. Thus, for both datasets, I considered different values of K to see if it improves my performance metric, which is the percentage of incorrectly clustered data. Here are the results from my implementation with varying Ks. As you can see from the plot, as K increases, the accuracy of the algorithm also steadily increases for both datasets. For the National Consensus Dataset, the optimal value for K was 4, and for the Letter Dataset, the optimal K was 6. This seems to make sense because in classification the algorithm will associate a cluster with a certain class.

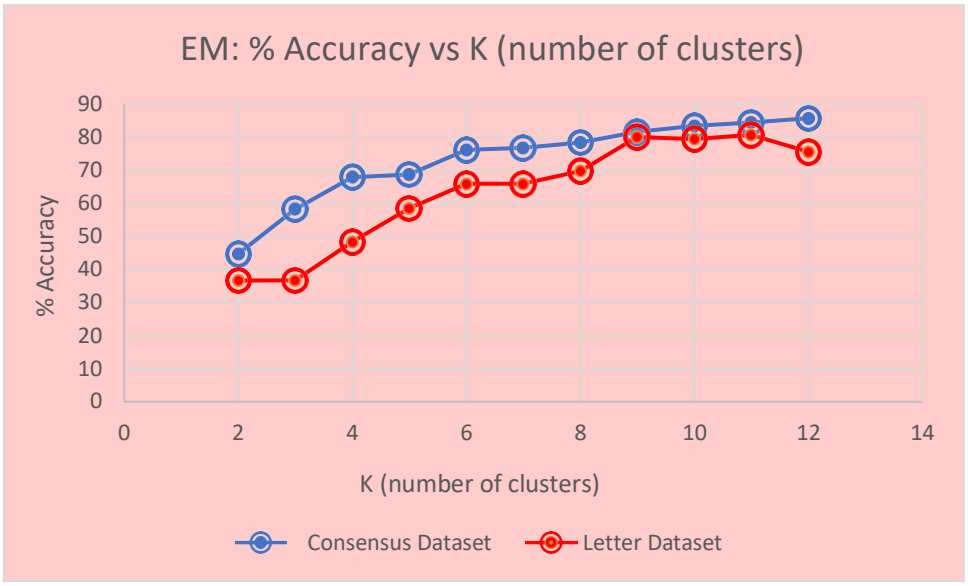


EM

The next algorithm I implemented was also a clustering one, EM, which uses iteration to alternate between performing an expectation step, which estimates based off current parameters,

and a maximization step, which computes parameters that maximizes the estimation in the E step. EM is different from K-means in that a point can be in potentially as many clusters as possible, whereas in K-means, a point is either in a cluster or not. Instead of a data point either being in the cluster or not, it's assigned a probability of being in a certain cluster, which represents the Gaussian distribution for each cluster. What I did for EM was first run WEKA's EM algorithm on the two dataset without manually adjusting anything, then, I compared this to running WEKA's EM algorithm and manually selecting the optimal K from K-means, which for my datasets were 4 and 6. The results are shown below and confirms what we already expected to happen. The manually selected K = 4 and K = 6 gave better clusters and results than the arbitrarily chosen K, which seemed to overfit the data. (below on the left is a representation of

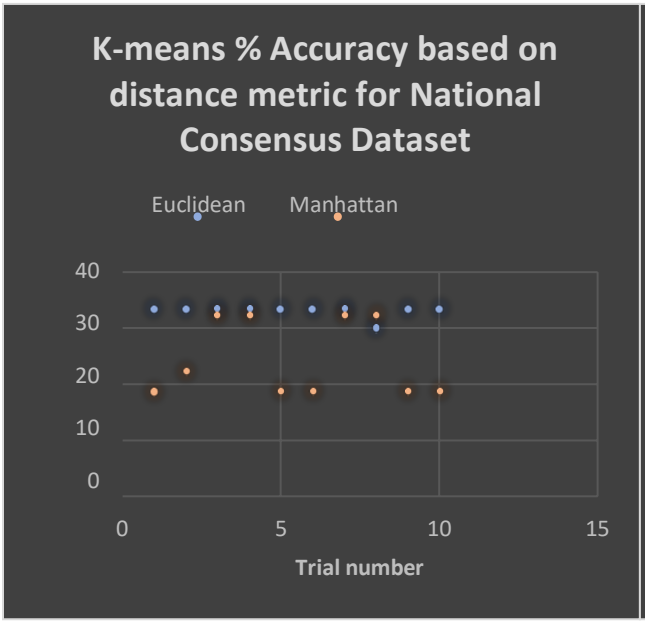
the K = 4 clusters for the National Consensus Dataset, on the right K = 6 for the Letters Dataset)

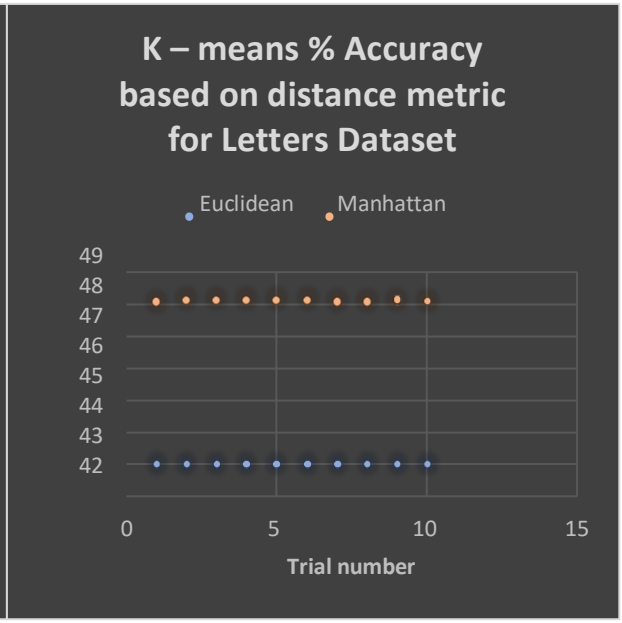


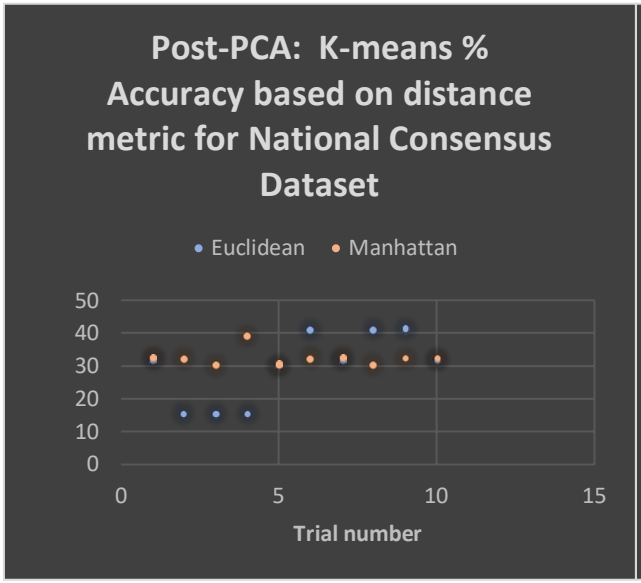
Principal Component Analysis (PCA):

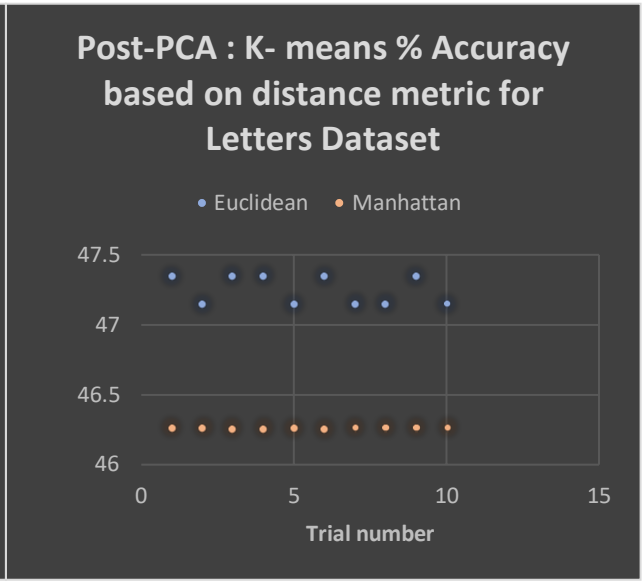
Principal Component Analysis is a feature transformation algorithm that basically transform the problem into an eigenproblem and then tries to find a linear transformation of the features. The linear transformation is then plotted as the axes of the data and the projections of the data onto those axes have maximum variance, and so they are principal component vectors. The more principal component vectors are found, the lesser the eigenvalues of each vector. As stated in my introduction, I will apply PCA to both datasets and compare the results of both the clustering algorithms (K-means and EM) after application of PCA to their results beforehand in the previous sections. In both problems, after applying PCA, the dimensionality of our data was reduced (as expected, as it is literally a dimensionality reduction algorithm) through taking linear combinations of the original data. The eigenvalues represent the dimensions of most variance from greatest to least.

For clustering after PCA, I ran multiple trials of each algorithm with a manually fixed K of 2 clusters. For the performance metric, I used Manhattan and Euclidean distance for both datasets. The plots for K-means before application of PCA vs After are shown below, with the application of performance metric of the Manhattan and Euclidean distance. From observation of the plots below, we can see that PCA actually made a solid difference in the clustering of both datasets, as the percent of incorrectly classified data became lower. Furthermore, the results for EM also saw steady improvement, with the National Consensus Dataset improving by 3% and the Letter Dataset improving by 2%.









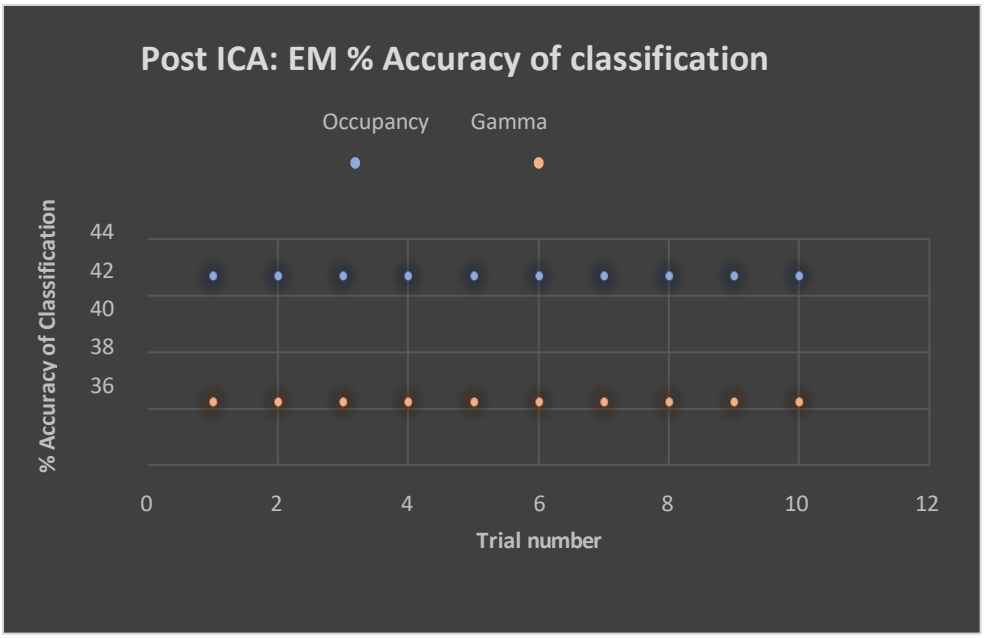
Independent Component Analysis (ICA)

Independent component analysis (ICA) is another algorithm that uses feature transformation to do dimensionality reduction. Independent component analysis assumes there are hidden features that are underlying a dataset's described features. It finds these independent

hidden features and derives independent components from them to try to use this and combine them to reconstruct the original dataset in a different format. For ICA, I ran WEKA's on both datasets and looked at the kurtosis of each component and compared it to a normal distribution. For ICA, maximizing the independence between components is closely related to maximizing their non-Gaussianity. Thus, to achieve this I will try to maximize the absolute or squared kurtosis, which is a way to measure this non-Gaussianity. One drawback of this method is that estimation of kurtosis is highly sensitive to outliers, so it could not be the best objective function for ICA, but I will give it a try anyway.

As the standard ICA model (such as the one I used on WEKA) already assumes the same number of sources as input dimensions, I did not have to do any adjustments. One way we could differ this in future experiments is to run PCA before ICA, but this could also cause the problem of reducing dimensionality too much, resulting in ICA fitting too many components.

When I tested the ICA generated data components with the clustering algorithms K-means and EM (using the same optimal value of K = 4 and 6), the results show a marginal improvement in the log likelihood. Also, if we just let WEKA arbitrarily choose K, it barely differentiated the results with if I had just done the clustering without any ICA.

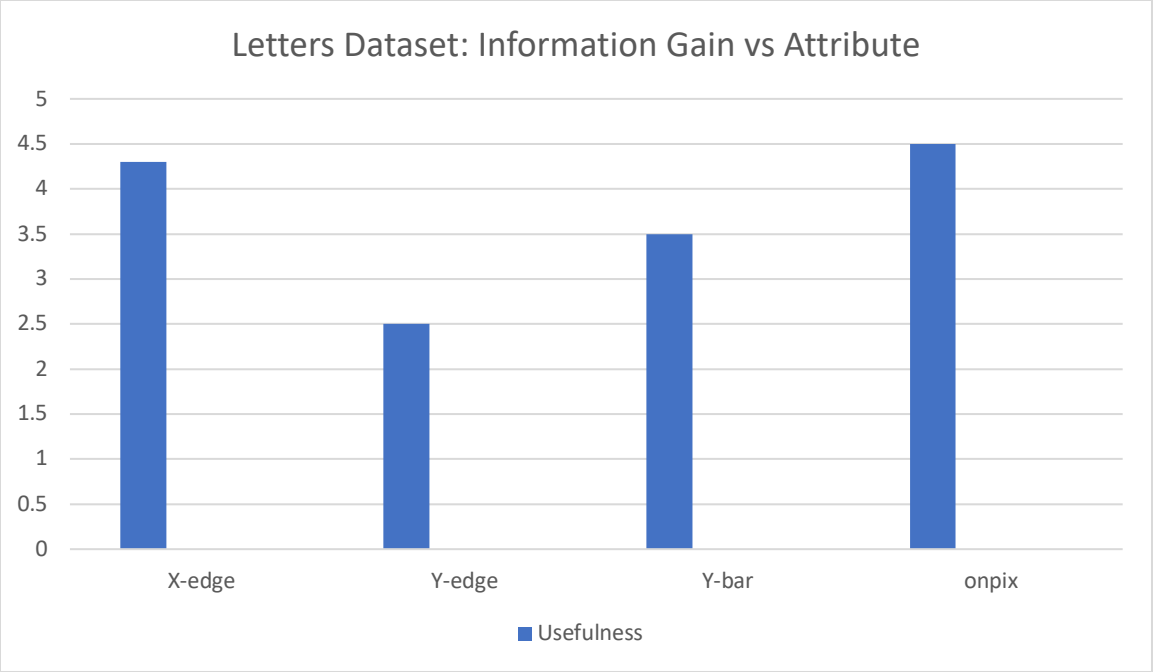


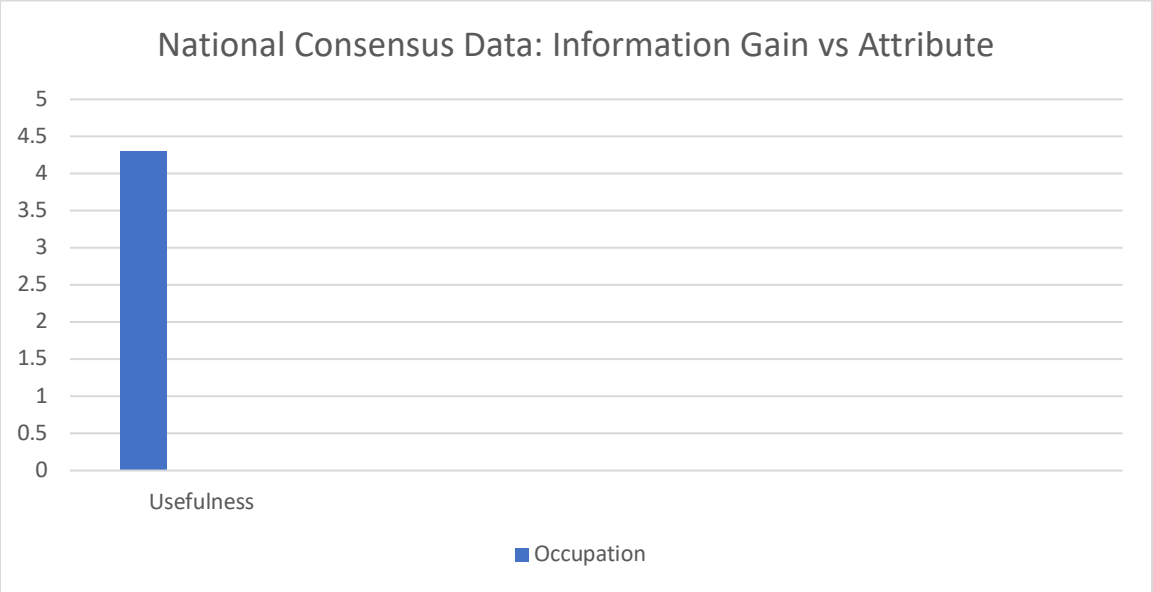
Randomized Projections (PR):

Randomized Projections is another dimensionality reducing algorithm that takes a target number of dimensions and uses this input to create a randomized matrix that is then used to transform and project the features of the original dataset into another feature vector with a smaller feature space, the feature space being the specified number of dimensions (although we could also specify a larger number of dimensions and thus a larger feature space). This is the hyperparameter we manipulate for Randomized projections. For the performance metric of this algorithm, I used clustering error, and ran the algorithm over several trials with randomized seeds, with my optimal K value of 4 and 6. I also decided to pick the target dimensions of each dataset as 4 and 6, for the National Consensus Dataset and the Letter Dataset, respectfully. For both datasets, the algorithm always settled between 4 and 7 clusters.

Information Gain (IG):

For my last algorithm, we were allowed to choose any feature selection algorithm to run, and since WEKA already has the information gain algorithm ID3, I chose that. Information gain is a Filter Feature Selection method, which means it applies a statistical measure to assign each feature a score that is meant to be representative of how useful the feature is in determining a class, or outcome. The features and then ranked by this assigned score and selected to either be kept or removed from the dataset. The methods are often univariate and consider the feature independently, or with regard to the dependent variable. I believe that there are no hyperparameters, but the algorithm ranks the usefulness of the various attributes of the two datasets, and the results of the rankings can be seen in the graphs attached below.


Letters Dataset: Information Gain vs Attribute
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
X-edge
Y-edge
Y-bar
onpix
Usefulness


National Consensus Data: Information Gain vs Attribute
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
Usefulness
Occupation

Conclusion:

To conclude the experiment, I analyzed the algorithms with a neural network. The results conclude that PCA, ICA, and Randomized Projections allow us to lessen the 'curse of dimensionality' and thus can bring better results in classification because less data is used to train and thus there is less overfitting. My neural network had 13 13 13 hidden layers, and a learning rate of 1E-5. The graphs below show that using PCA allowed us to have a 90% accuracy on the validation set after 200 epochs, which means PCA helped to make use of variance to achieve better results than without dimensionality reduction.

In summary, I have learned that applying different performance metrics to problems will affect their accuracy and results, and applying different dimensionality reduction algorithms in combination with clustering algorithms to reduce feature space can provide us with better results.

References

"Independent Component Analysis." *Wikipedia*, Wikimedia Foundation, 15 Mar. 2019, en.wikipedia.org/wiki/Independent_component_analysis.

"Principal Component Analysis." *Wikipedia*, Wikimedia Foundation, 26 Mar. 2019, en.wikipedia.org/wiki/Principal_component_analysis.

"Random Projection." *Wikipedia*, Wikimedia Foundation, 16 Feb. 2019, en.wikipedia.org/wiki/Random_projection.