Exploring Feature Reduction Techniques for Indic Script Text Clustering.

A project report submitted in partial fulfilment of the requirements For the award of the Degree of

Bachelor of Technology

in

Computer Science and Engineering

By

Abhay .A. Bhat (**16011A0501**),
Priyatosh Tripathy(**16011A0521**),
V. Sai Nagendra (**16011A0524**)

Under the guidance of

Prof. Dr. B. Padmaja Rani

Department of Computer Science and Engineering

Jawaharlal Nehru Technological University Hyderabad,

College of Engineering, Hyderabad – 500085
2019

Department of Computer Science and Engineering

Jawaharlal Nehru Technological University Hyderabad,

College of Engineering, Hyderabad – 500085

DECLARATION

We, Abhay.A.Bhat(16011A0501), Priyatosh Tripathy(16011A0521), V.Sai Nagendra(16011A0524) here by declare that the project report entitled “Exploring Feature Reduction Techniques for Indic Script Text Clustering.” carried out under the guidance of Dr. B. Padmaja Rani, is submitted in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering. This is a record of bonafide work carried out by us and the results embodied in this project have not been reproduced/copied from any source and have not been submitted to any other University or Institute for the award of any other degree or diploma.

Abhay.A.Bhat(**16011A0501**),
Priyatosh Tripathy(**16011A0521**),
V. Sai Nagendra(**16011A0524**).
Department of Computer Science and Engineering,
JNTUH College of Engineering,
Hyderabad.

Date:

Department of Computer Science and Engineering

Jawaharlal Nehru Technological University Hyderabad,

College of Engineering, Hyderabad – 500085

CERTIFICATE BY THE SUPERVISOR

This is to certify that the project report entitled “**Exploring Feature Reduction Techniques for Indic Script Text Clustering.**”, being submitted by Abhay.A.Bhat(**16011A0501),** Priyatosh Tripathy(**16011A0521),** V.Sai Nagendra(**16011A0524**) in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering, is a record of bonafide work carried out by them. The results are verified and found satisfactory.

Dr.B. Padmaja Rani,
Professor,
Department of Computer Science and Engineering,
JNTUH College of Engineering,
Hyderabad.

Date:

Department of Computer Science and Engineering

Jawaharlal Nehru Technological University Hyderabad,

College of Engineering, Hyderabad – 500085

CERTIFICATE BY THE HEAD OF THE DEPARTMENT

This is to certify that the project report entitled “**Exploring Feature Reduction Techniques for Indic Script Text Clustering**”, being submitted by Abhay.A.Bhat(**16011A0501),** Priyatosh Tripathy(**16011A0521),** V.Sai Nagendra(**16011A0524**), in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering, is a record of bonafide work carried out by them. The results are verified and found satisfactory.

Dr.R. Sri Devi,
Professor & Head of the Department,
Department of Computer Science and Engineering,
JNTUH College of Engineering,
Hyderabad.
Date:

ACKNOWLEDGEMENT

We would like to express sincere thanks to our Supervisor Dr. B. Padmaja Rani ma’am, Professor for her admirable guidance and inspiration both theoretically and practically and most importantly for the drive to complete project successfully. Working under such an eminent guide was our privilege.

We owe a debt of gratitude to Dr. R. SriDevi ma’am, Professor & **Head of the department of Computer Science & Engineering**, for her kind considerations and encouragement in carrying out this project successfully.

We are grateful to the Project Review Committee members and Department of Computer Science & Engineering who have helped in successfully completing this project by giving their valuable suggestions and support.

We express thanks to our parents for their love, care and moral support without which we would have not been able to complete this project. It has been a constant source of inspiration for all our academic endeavours. Last but not the least, we thank the Almighty for making us a part of the world.

Abhay. A. Bhat(16011A0501)
Priyatosh Tripathy(16011A0521)
V. Sai Nagendra(16011A0524)

ABSTRACT

Increasingly large text datasets and the high dimensionality associated with natural language especially Indic languages create a great challenge in text mining as important information is usually lost in the mining process. In this project a systematic study is conducted, in which document representation methods are used in combination with different Dimensionality Reduction Techniques, in the context of Telugu Text Clustering problem. The datasets used are of different subject matter such as Literature, Sports, Politics, News etc.

The dimensionality reduction methods include Principal Component Analysis (PCA), Independent Component Analysis (ICA), Kernel Principal Component Analysis (KPCA). Results are compared in terms of clustering performance, using the k-means clustering algorithm. This entire project is to be done using python.

Abhay.A.Bhat (16011A0501)
Priyatosh Tripathy (16011A0521)
V. Sai Nagendra (16011A0524)

TABLE OF CONTENTS :

1. INTRODUCTION

1.1 Text Processing:

Text processing has a direct application to Natural Language Processing, also known as NLP. NLP is aimed at processing the languages spoken or written by humans when they communicate with one another. This is different from the communication between a computer and a human where the communication is wither a computer program written by human or some gesture by human like clicking the mouse at some position. NLP tries to understand the natural language spoken by humans and classify it, analyses it as well if required respond to it. Python has a rich set of libraries which cater to the needs of NLP. The Natural Language ToolKit (NLTK) is a suite of such libraries which provides the functionalities required for NLP.

1.2 Machine Learning

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions according

Machine learning algorithms are often categorized as supervised or unsupervised.

1.2.1 Supervised Learning

Supervised machine learning algorithms can apply what has been learned in the past to new data using labelled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.

1.2.2 Unsupervised Learning

In contrast, unsupervised machine learning algorithms are used when the information used to train is neither classified nor labelled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabelled data. The system doesn’t figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabelled data.

1.2.3. Semi-supervised Learning

Semi-supervised machine learning algorithms fall somewhere in between supervised and unsupervised learning, since they use both labelled and unlabelled data for training – typically a small amount of labelled data and a large amount of unlabelled data. The systems that use this method are able to considerably improve learning accuracy. Usually, semi-supervised learning is chosen when the acquired labelled data requires skilled and relevant resources in order to train it or learn from it. Otherwise, acquiring unlabelled data generally doesn’t require additional resources.

1.2.4 Reinforcement Learning

Reinforcement machine learning algorithms is a learning method that interacts with its environment by producing actions and discovers errors or rewards. Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behaviour within a specific context in order to maximize its performance. Simple reward feedback is required for the agent to learn which action is best; this is known as the reinforcement signal.

1.3 Natural Language Processing:

Natural language processing (NLP) is the ability of a computer program to understand human language as it is spoken. NLP is a component of artificial intelligence. The development of NLP applications is challenging because computers traditionally require humans to "speak" to them in a programming language that is precise, unambiguous and highly structured, or through a limited number of clearly enunciated voice commands. Human speech, however, is not always precise -- it is often ambiguous and the linguistic structure can depend on many complex variables, including slang, regional dialects and social context.

Most of the research being done on natural language processing revolves around search, especially enterprise search. This involves allowing users to query data sets in the form of a question that they might pose to another person. The machine interprets the important elements of the human language sentence, such as those that might correspond to specific features in a data set, and returns an answer.

NLP can be used to interpret free text and make it analyzable. There is a tremendous amount of information stored in free text files, like patients' medical records, for example. Prior to deep learning-based NLP models, this information was inaccessible to computer-assisted analysis and could not be analyzed in any kind of systematic way. But NLP allows analysts to sift through massive troves of free text to find relevant information in the files.

Sentiment analysis is another primary use case for NLP. Using sentiment analysis, data scientists can assess comments on social media to see how their business's brand is performing, for example, or review notes from customer service teams to identify areas where people want the business to perform better.

Google and other search engines base their machine translation technology on NLP deep learning models. This allows algorithms to read text on a webpage, interpret its meaning and translate it to another language.

1.5 Clustering

It is basically a type of unsupervised learning method . An unsupervised learning method is a method in which we draw references from datasets consisting of input data without labeled responses. Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of examples.

Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

1.5.1 Clustering Methods :

1. Density-Based Methods : These methods consider the clusters as the dense region having some similarity and different from the lower dense region of the space. These methods have good accuracy and ability to merge two clusters.Example DBSCAN (Density-Based Spatial Clustering of Applications with Noise) , OPTICS (Ordering Points to Identify Clustering Structure) etc.

2. Hierarchical Based Methods : The clusters formed in this method forms a tree type structure based on the hierarchy. New clusters are formed using the previously formed one. It is divided into two category

→ Agglomerative (bottom up approach)

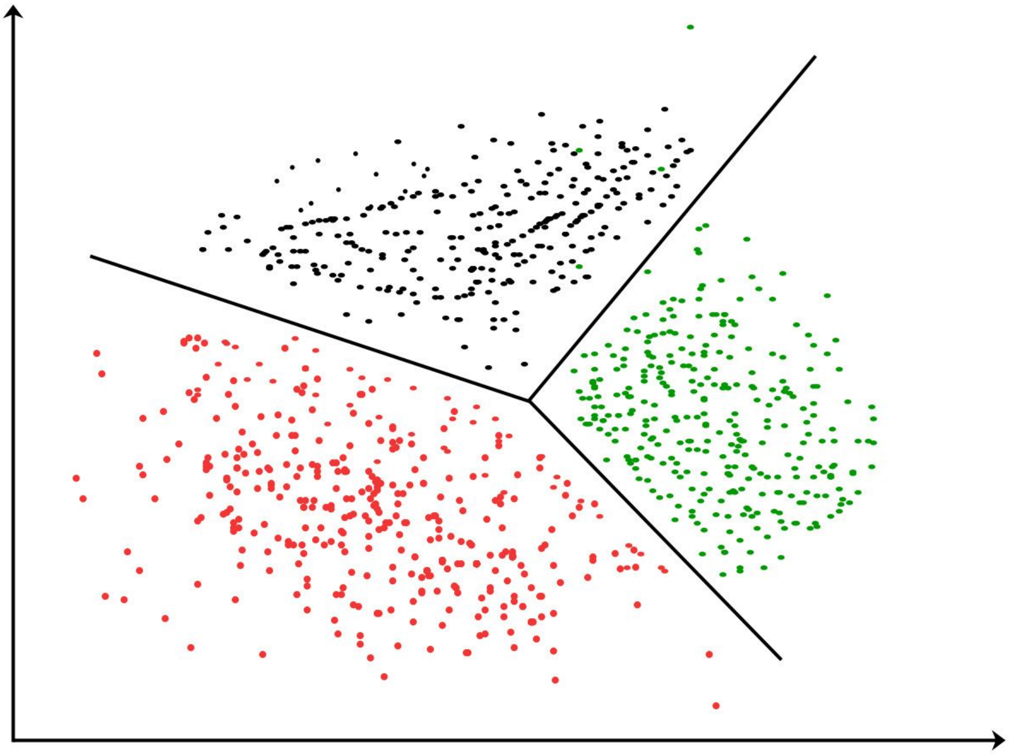
→ Divisive (top down approach) .

3. Partitioning Methods : These methods partition the objects into k clusters and each partition forms one cluster. This method is used to optimize an objective criterion similarity function such as when the distance is a major parameter example K-means, CLARANS (Clustering Large Applications based upon randomized Search) etc.

4. Grid-based Methods : In this method the data space are formulated into a finite number of cells that form a grid-like structure. All the clustering operation done on these grids are fast and independent of the number of data objects example STING (Statistical Information Grid), wave cluster, CLIQUE (CLustering In Quest) etc.

1.5.2 Clustering Algorithms :

K-means clustering algorithm – It is the simplest unsupervised learning algorithm that solves clustering problem.K-means algorithm partition n observations into k clusters where each observation belongs to the cluster with the nearest mean serving as a prototype of the cluster .



1.6 Cluster Evaluation

Clustering is an unsupervised machine learning algorithm. It helps in clustering data points to groups. Validating the clustering algorithm is bit tricky compared to supervised machine learning algorithm as clustering process does not contain ground truth labels. If one want to do clustering with ground truth labels being present, validation methods and metrics of supervised machine learning algorithms can be used.

1.6.1 Adjusted Rand Index

The Rand index or Rand measure (named after William M. Rand) in statistics, and in particular in data clustering, is a measure of the similarity between two data clusterings. A form of the Rand index may be defined that is adjusted for the chance grouping of elements, this is the adjusted Rand index. From a mathematical standpoint, Rand index is related to the accuracy, but is applicable even when class labels are not used.

Given a set of n elements **S** = { $\mathbf{o_1, o_2, o_3...}$} and two partitions to compare, **X** = {$\mathbf{X_1, X_2... X_r}$}, a partition of S into r subsets, and **Y** = {$\mathbf{Y_1, Y_2...Y_s}$}, a partition of S into s subsets, define the following:

a, the number of pairs of elements in **S** that are in the same subset in **X** and in same subset in **Y**.

b, the number of pairs of elements in **S** that are in the different subset in **X** and in different subset in **Y**.

c, the number of pairs of elements in **S** that are in the same subset in **X** and in different subset in **Y**.

d, the number of pairs of elements in **S** that are in the different subset in **X** and in same subset in **Y**.

The rand index is given by

$$R = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}}$$

Or

$$RI = \frac{TP+TN}{TP+FP+FN+TN}$$

The Adjusted Rand is given by the permutation model

$$\overbrace{ARI}^{\text{Adjusted Index}} = \frac{\sum_{ij}\binom{n_{ij}}{2} - [\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}]/\binom{n}{2}}{\frac{1}{2}[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}] - [\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}]/\binom{n}{2}}$$

Advantages:

- Random (uniform) label assignments have a ARI score close to 0.0 for any value of n_clusters and n_samples (which is not the case for raw Rand index or the V-measure for instance).
- Bounded range [-1, 1]: negative values are bad (independent labelings), similar clusterings have a positive ARI, 1.0 is the perfect match score.
- No assumption is made on the cluster structure: can be used to compare clustering algorithms such as k-means which assumes isotropic blob shapes with results of spectral clustering algorithms which can find cluster with “folded” shapes.

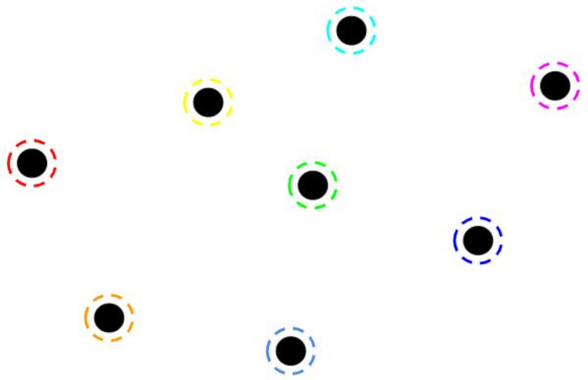
Drawbacks:

- ARI requires knowledge of the ground truth classes while is almost never available in practice or requires manual assignment by human annotators (as in the supervised learning setting).

1.6.2 Homogeneity

A perfectly homogeneous clustering is one where each cluster has data-points belonging to the same class label. Homogeneity describes the closeness of the clustering algorithm to this perfection.

Trivial Homogeneity: It is the case when the number of clusters is equal to the number of data points and each point is in exactly one cluster. It is the extreme case when homogeneity is highest while completeness is minimum.



Homogeneity is given by

$$h = 1 - \frac{H(C,K)}{H(C)}$$

where

$$H(C,K) = -\sum_{k=1}^{K}\sum_{c=1}^{C}\frac{a_{ck}}{N}\log(\frac{a_{ck}}{\sum_{c=1}^{C}a_{ck}})$$

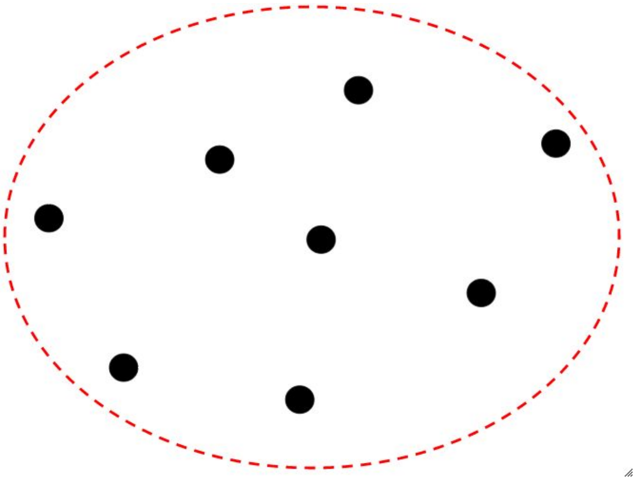
and

$$H(C) = -\sum_{c=1}^{C}\frac{\sum_{k=1}^{K}a_{ck}}{C}\log(\frac{\sum_{k=1}^{K}a_{ck}}{C})$$

1.6.3 Completeness

A perfectly complete clustering is one where all data-points belonging to the same class are clustered into the same cluster. Completeness describes the closeness of the clustering algorithm to this perfection.

Trivial Completeness: It is the case when all the data points are clustered into one cluster. It is the extreme case when homogeneity is minimum and completeness is maximum**.**



completeness is given by:

$$c = 1 - \frac{H(K,C)}{H(K)}$$

where

$$H(K,C) = -\sum_{c=1}^{C}\sum_{k=1}^{K}\frac{a_{ck}}{N}log(\frac{a_{ck}}{\sum_{k=1}^{K}a_{ck}})$$

1.6.4 V-measure

V-measure is calculated as the harmonic mean of homogeneity and completeness.

$$V_\beta = \frac{(1+\beta)hc}{\beta h+c}$$

Where factor β can be adjusted to favour either the homogeneity or the completeness of the clustering algorithm.

Advantages:

- Bounded scores: 0.0 is as bad as it can be, 1.0 is a perfect score.
- Intuitive interpretation: clustering with bad V-measure can be qualitatively analyzed in terms of homogeneity and completeness to better feel what ‘kind’ of mistakes is done by the assignment.

- No assumption is made on the cluster structure: can be used to compare clustering algorithms such as k-means which assumes isotropic blob shapes with results of spectral clustering algorithms which can find cluster with “folded” shapes.

Drawbacks:

- The previously introduced metrics are not normalized with regards to random labeling: this means that depending on the number of samples, clusters and ground truth classes, a completely random labeling will not always yield the same values for homogeneity, completeness and hence v-measure. In particular random labeling won’t yield zero scores especially when the number of clusters is large.

2. LITERATURE

2.1 Text Clustering:

Text clustering is the application of cluster analysis to text-based documents. It uses machine learning and natural language processing (NLP) to understand and categorize unstructured, textual data.

Typically, descriptors (sets of words that describe topic matter) are extracted from the document first. Then they are analyzed for the frequency in which they are found in the document compared to other terms. After which, clusters of descriptors can be identified and then auto-tagged.

From there, the information can be used in any number of ways. Google’s search engine is probably the best and most widely known example. When you search for a term on Google, it pulls up pages that apply to that term.

2.1.1 Application of text clustering:

1. Automatic document organization.
2. Topic extraction.
3. Information retrieval.
4. Document Filtering.

2.1.2 Problems with indic languages:

1. English is known to be an Internet’s number one language. Accuracy has improved from 11% - 97.50% within the past decade.
2. Morphology is richer and word order is flexible.

3. Indic languages are highly inflective.

2.2 Document Representation:

Document representation is concerned about how textual documents should be represented in various tasks, e.g. text processing, retrieval and knowledge discovery and mining. Its prevailing approach is the vector space model, i.e. a document di is represented as a vector of term weights , where is the collection of terms that occur at least once in the document collection D.

2.2.1 TF-IDF Model

tf-idf stands for Term frequency-inverse document frequency. The tf-idf weight is a weight often used in information retrieval and text mining. Variations of the tf-idf weighting scheme are often used by search engines in scoring and ranking a document's relevance given a query. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus (data-set).

2.2.1.1 Method of computation

tf-idf is a weighting scheme that assigns each term in a document a weight based on its term frequency (tf) and inverse document frequency (idf). The terms with higher weight scores are considered to be more important.

Typically, the tf-idf weight is composed by two terms-
Normalized Term Frequency (tf)
Inverse Document Frequency (idf)

Let’s us take 3 documents to show how this works.

Doc 1: Ben studies about computers in Computer Lab.
Doc 2: Steve teaches at Brown University.
Doc 3: Data Scientists work on large datasets.

Let’s say we are doing a search on these documents with the following query: Data Scientists

The query is a free text query. It means a query in which the terms of the query are typed freeform into the search interface, without any connecting search operators.

Step 1: Computing the Term Frequency(tf)
Frequency indicates the number of occurences of a particular term t in document d. Therefore,

tf(t, d) = N(t, d), wherein tf(t, d) = term frequency for a term t in document d.
N(t, d) = number of times a term t occurs in document d

We can see that as a term appears more in the document it becomes more important, which is logical.We can use a vector to represent the document in bag of words model, since the ordering of terms is not important. There is an entry for each unique term in the document with the value being its term frequency.

tf(t, d) = N(t, d) / ||D||
wherein, ||D|| = Total number of term in the document

Step 2: Compute the Inverse Document Frequency – idf

It typically measures how important a term is. The main purpose of doing a search is to find out the relevant documents matching the query. Since tf considers all terms equally important, thus, we can't only use term frequencies to calculate the weight of a term in the document. However, it is known that certain terms, such as "is", "of", and "that" may appear a lot of times but have little importance. Thus we need to weigh down the frequent terms while scaling up the rare ones. Logarithms helps us to solve this problem.

First of all, find the document frequency of a term t by counting the number of documents containing the term:

$df(t) = N(t)$

where-

$df(t)$ = Document frequency of a term t

$N(t)$ = Number of documents containing the term t

Term frequency is the occurrence count of a term in one particular document only; while document frequency is the number of different documents the term appears in, so it depends on the whole corpus. Now let's look at the definition of inverse document frequency. The idf of a term is the number of documents in the corpus divided by the document frequency of a term.

$idf(t) = N/ df(t) = N/N(t)$

It's expected that the more frequent term to be considered less important, but the factor (most probably integers) seems too harsh. Therefore, we take the logarithm (with base 2) of the inverse document frequencies. So, the idf of a term t becomes :

$idf(t) = \log(N/ df(t))$

This is better, and since log is a monotonically increasing function we can safely use it.

Step 3: tf-idf Scoring

Now we have defined both tf and idf and now we can combine these to produce the ultimate score of a term t in document d. Therefore,

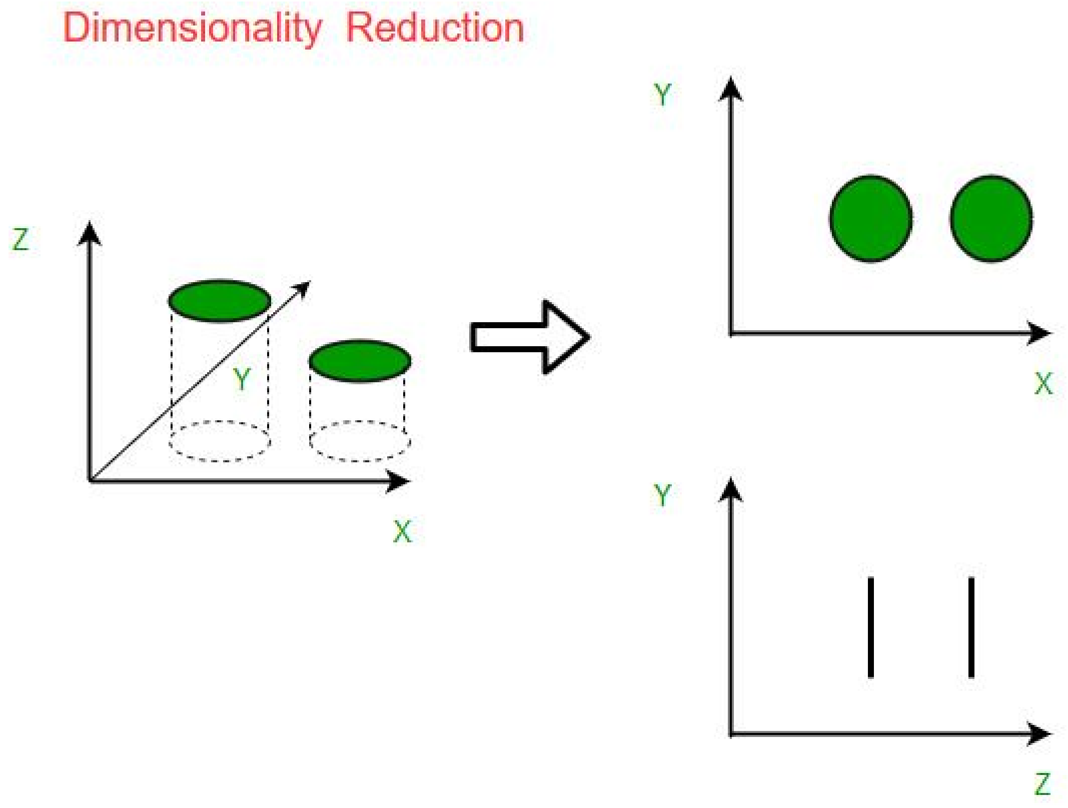
tf-idf(t, d) = tf(t, d)* idf(t, d)

2.3 Feature Extraction and Dimensionality reduction

Advances in data collection and storage capabilities during the past decades have led to an information overload in most sciences. Researchers working in domains as diverse as engineering, astronomy, biology, remote sensing, economics, and consumer transactions, face larger and larger observations and simulations on a daily basis. Such datasets, in contrast with smaller, more traditional datasets that have been studied extensively in the past, present new challenges in data analysis. Traditional statistical methods break down partly because of the increase in the number of observations, but mostly because of the increase in the number of variables associated with each observation.

The dimension of the data is the number of variables that are measured on each observation. High-dimensional datasets present many mathematical challenges as well as opportunities, and are bound to give rise to new theoretical developments. One of the problems with high-dimensional datasets is that, in many cases, not all the measured variables are important" for understanding the underlying phenomena of interest. While certain computationally expensive novel methods can construct predictive models with high accuracy from high-dimensional data, it is still of interest in many applications to reduce the dimension of the original data prior to any modelling of the data.

High dimensionality reduction solves the problem of inefficient computation and increase the difficulty in detecting and exploiting the relationships among terms. After getting the relationship among the key words, clustering is done very effectively and easily. The processing time will be reduced.



2.3.1 Principal Component Analysis

Principal Component Analysis is basically a statistical procedure to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables.
Each of the principal components is chosen in such a way so that it would describe most of the still available variance and all these principal components are orthogonal to each other. In all principal components first principal component has maximum variance.

Uses of PCA:

1. It is used to find inter-relation between variables in the data.
2. It is used to interpret and visualize data.
3. As the number of variables are decreasing it makes further analysis simpler.
4. It’s often used to visualize genetic distance and relatedness between populations.
5. These are basically performed on square symmetric matrix. It can be a pure sums of squares and cross products matrix or Covariance matrix or Correlation matrix. A correlation matrix is used if the individual variance differs much.

Objectives of PCA:

1. It is basically a non-dependent procedure in which it reduces attribute space from a large number of variables to a smaller number of factors.
2. PCA is basically a dimension reduction process but there is no guarantee that the dimension is interpretable.
3. Main task in this PCA is to select a subset of variables from a larger set, based on which original variables have the highest correlation with the principal amount.

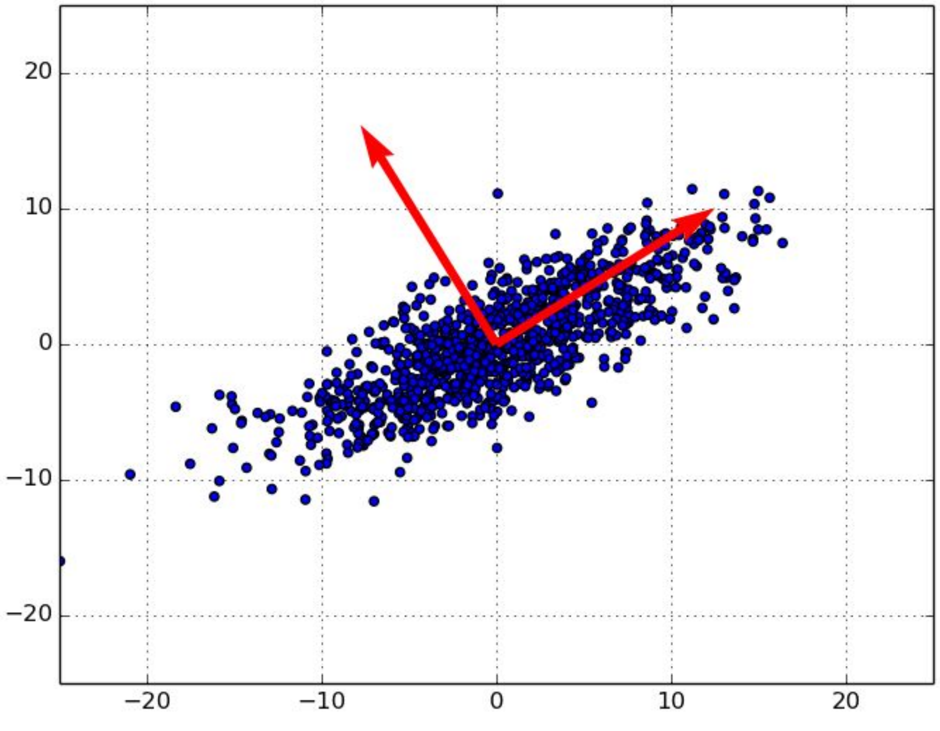
Principal Axis Method:

PCA basically search a linear combination of variables so that we can extract maximum variance from the variables. Once this process completes it removes it and search for another linear combination which gives an explanation about the maximum proportion of remaining variance which basically leads to orthogonal factors. In this method, we analyze total variance.

Eigenvector: It is a non-zero vector that stays parallel after matrix multiplication. Let’s suppose x is eigen vector of dimension r of matrix M with dimension r*r if Mx and x are parallel. Then we need to solve Mx=Ax where both x and A are unknown to get eigenvector and eigenvalues.

Under Eigen-Vectors we can say that Principal components show both common and unique variance of the variable. Basically, it is variance focused approach seeking to reproduce total variance and correlation with all components. The principal components are basically the linear combinations of the original variables weighted by their contribution to explain the variance in a particular orthogonal dimension.

Eigenvalues: It is basically known as characteristic roots. It basically measures the variance in all variables which is accounted for by that factor. The ratio of eigenvalues is the ratio of explanatory importance of the factors with respect to the variables. If the factor is low then it is contributing less in explanation of variables. In simple words, it measures the amount of variance in the total given database accounted by the factor. We can calculate the factor’s eigenvalue as the sum of its squared factor loading for all the variables.

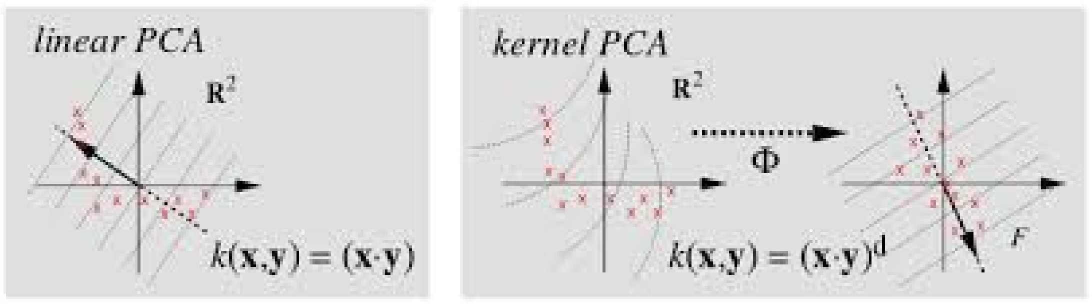


2.3.2 Kernel Principal Component Analysis:

PCA is a linear method. That is it can only be applied to datasets which are linearly separable. It does an excellent job for datasets, which are linearly separable. But, if we use it to non-linear datasets, we might get a result which may not be the optimal dimensionality

reduction. Kernel PCA uses a kernel function to project dataset into a higher dimensional feature space, where it is linearly separable. It is similar to the idea of Support Vector Machines.
There are various kernel methods like linear, polynomial, and gaussian.

Kernel PCA uses a kernel function to project dataset into a higher dimensional feature space, where it is linearly separable. It is similar to the idea of **Support Vector Machines**.



2.4 Packages

Sklearn:
Scikit-learn is an open source Python library that implements a range of machine learning, pre-processing, cross-validation and visualization algorithms using a unified interface.
Important features of scikit-learn:
•Simple and efficient tool for data mining and analysis. It features various classification, regression and clustering algorithms including support vector machines and random forests, gradient boosting, k-means, etc.
•Built on top of NumPy, SciPy, and matplotlib

Functions used:

- Document representation:
  sklearn.feature_extraction.text.TfidfVectorizer
- Clustering:
  sklearn.cluster.KMeans
- Dimensionality Reduction:
  sklearn.decomposition.PCA
  sklearn.decomposition. KernelPCA

sklearn.decomposition. TruncatedSVD

- Cluster evaluation:
  sklearn.metrics.adjusted_rand_score
  sklearn.metrics.v_measure_score
  sklearn.metrics.homogeneity_score

3. IMPLEMENTATION

3.1 Dataset:

This is a set of Telugu documents from various sources belonging to one of rivers, news, temples, etc.

Link to the dataset:
https://github.com/Ray784/mini_project/tree/master/TeluguDocuments

3.2 Preprocessing

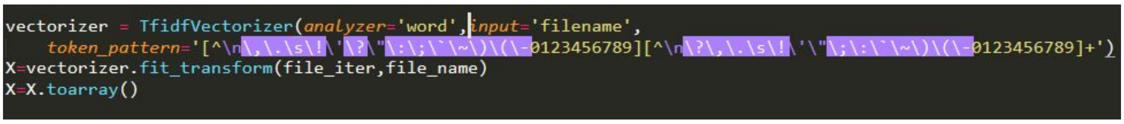
Since, text is the most unstructured form of all the available data, various types of noise are present in it and the data is not readily analyzable without any pre-processing. The entire process of cleaning and standardization of text, making it noise-free and ready for analysis is known as text pre-processing.

Noise removal:
Processing of indic script is difficult hence noise removal consists of removal of numbers and punctuation.

Stemming is also difficult as telugu us highly inflictive i.e the words are combined to produce a meaning. Prefix and suffix removal may change the meaning.

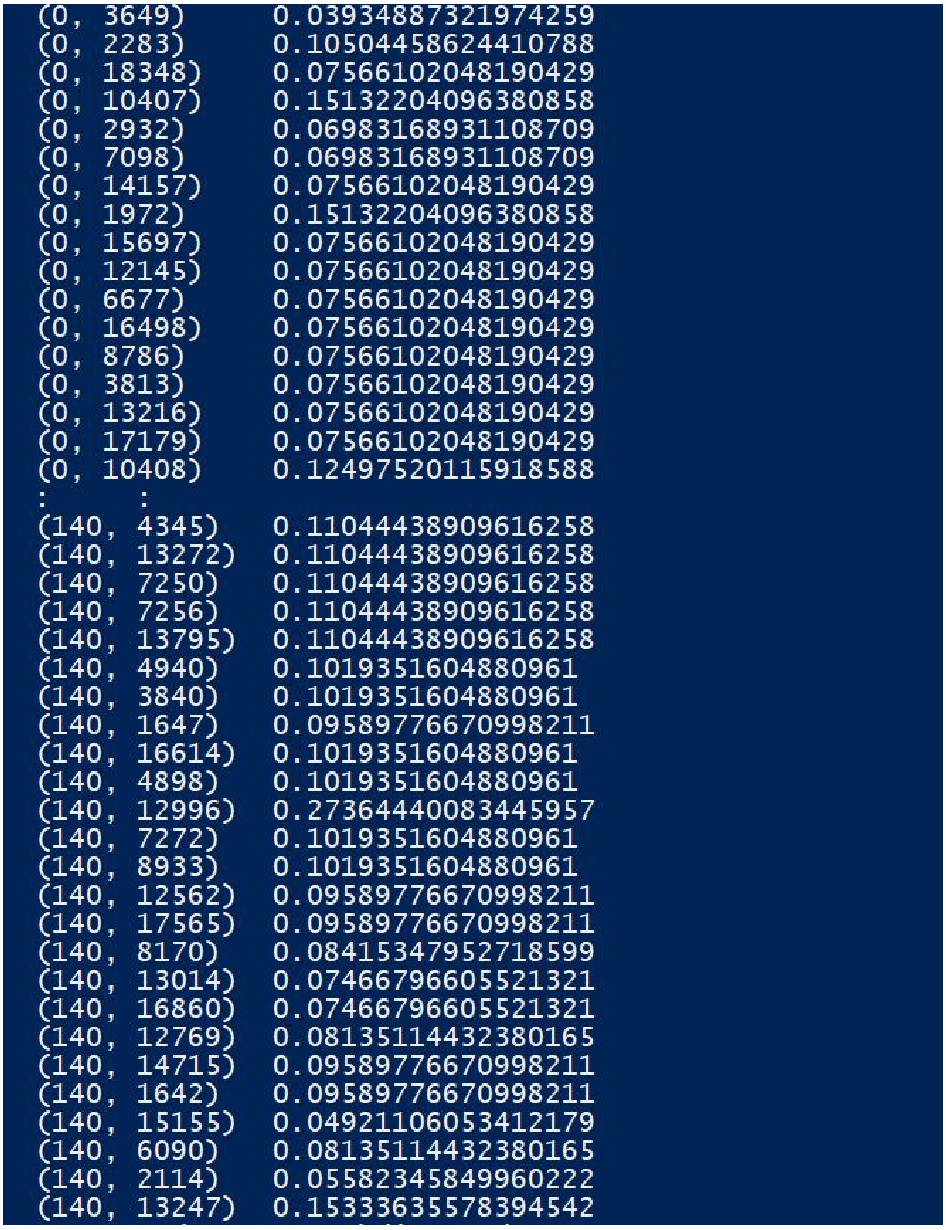
3.3 Document Representation:

sklearn.feature_extraction.text.TfidfVectorizer from the sklearn package of python is used for converting unstructured data into vectors with unique words as dimensions.



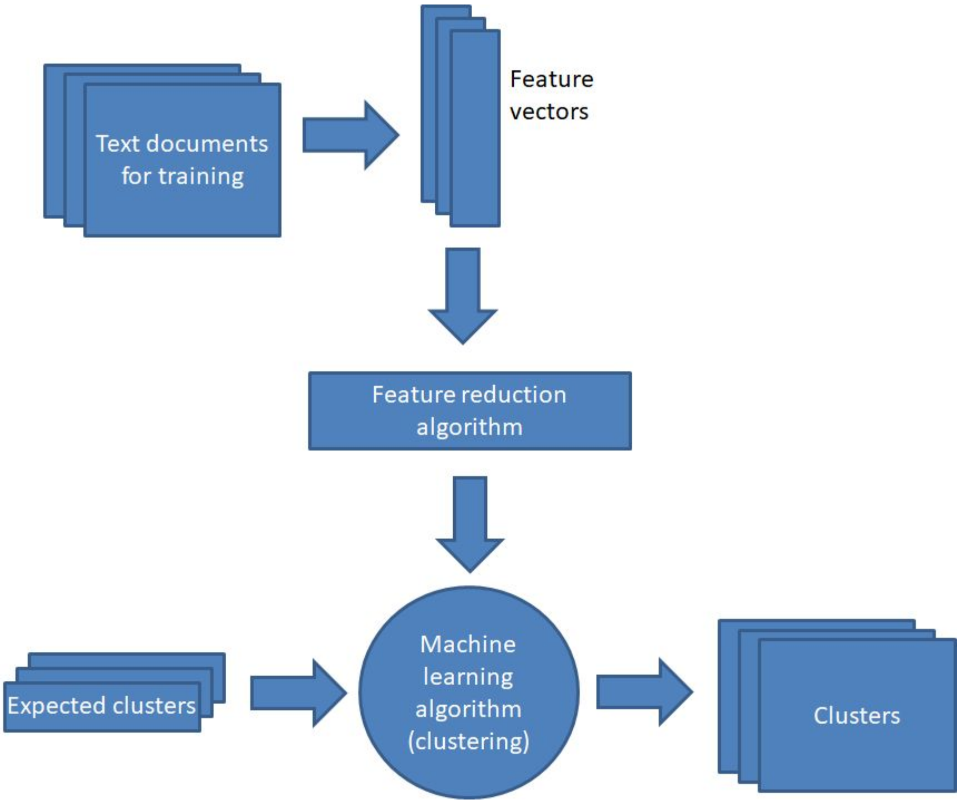
```
vectorizer = TfidfVectorizer(analyzer='word',input='filename',
    token_pattern='[^\n\,\.\s\!\'\?\"\:\;\`\~\)\(\-0123456789][^\n\?\,\.\s\!\'\"\;\:\`\~\)\(\-0123456789]+')
X=vectorizer.fit_transform(file_iter,file_name)
X=X.toarray()
```

Tf-IDF vectoriser output:(sparse matrix):



```
  (0, 3649)     0.03934887321974259
  (0, 2283)     0.10504458624410788
  (0, 18348)    0.07566102048190429
  (0, 10407)    0.15132204096380858
  (0, 2932)     0.06983168931108709
  (0, 7098)     0.06983168931108709
  (0, 14157)    0.07566102048190429
  (0, 1972)     0.15132204096380858
  (0, 15697)    0.07566102048190429
  (0, 12145)    0.07566102048190429
  (0, 6677)     0.07566102048190429
  (0, 16498)    0.07566102048190429
  (0, 8786)     0.07566102048190429
  (0, 3813)     0.07566102048190429
  (0, 13216)    0.07566102048190429
  (0, 17179)    0.07566102048190429
  (0, 10408)    0.12497520115918588
  :     :
  (140, 4345)   0.11044438909616258
  (140, 13272)  0.11044438909616258
  (140, 7250)   0.11044438909616258
  (140, 7256)   0.11044438909616258
  (140, 13795)  0.11044438909616258
  (140, 4940)   0.1019351604880961
  (140, 3840)   0.1019351604880961
  (140, 1647)   0.09589776670998211
  (140, 16614)  0.1019351604880961
  (140, 4898)   0.1019351604880961
  (140, 12996)  0.27364440083445957
  (140, 7272)   0.1019351604880961
  (140, 8933)   0.1019351604880961
  (140, 12562)  0.09589776670998211
  (140, 17565)  0.09589776670998211
  (140, 8170)   0.08415347952718599
  (140, 13014)  0.07466796605521321
  (140, 16860)  0.07466796605521321
  (140, 12769)  0.08135114432380165
  (140, 14715)  0.09589776670998211
  (140, 1642)   0.09589776670998211
  (140, 15155)  0.04921106053412179
  (140, 6090)   0.08135114432380165
  (140, 2114)   0.05582345849960222
  (140, 13247)  0.15333635578394542
```

3.4 Training

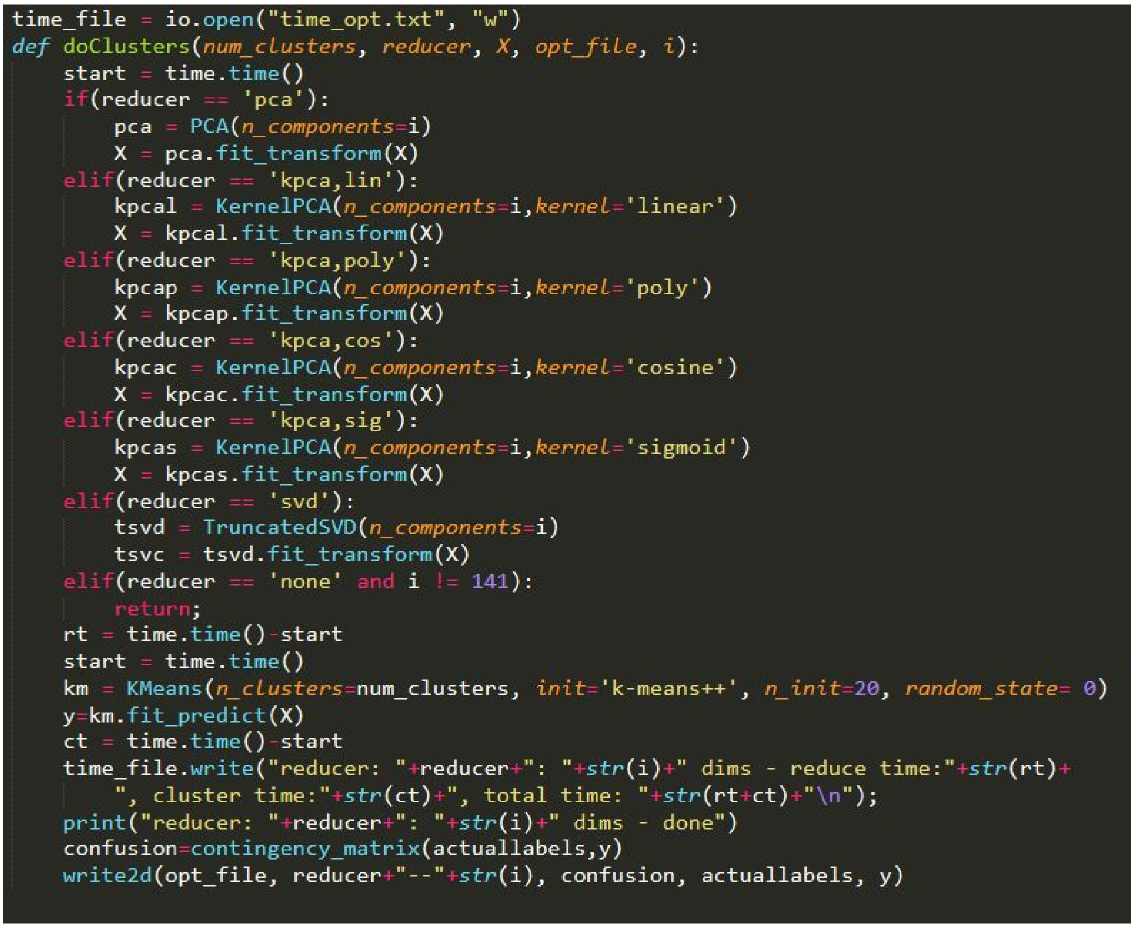

Text documents for training
Feature vectors
Feature reduction algorithm
Expected clusters
Machine learning algorithm (clustering)
Clusters

3.4.1 Feature reduction

PCA, Kernel PCA(linear, polynomial, cosine and sigmoid) and SVD are used for feature reduction.

K-Means is used for clustering.

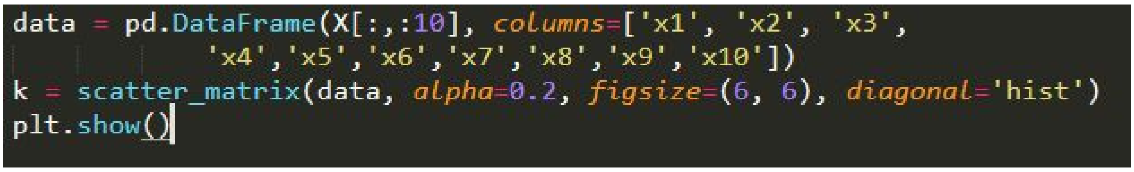
sklearn.decomposition.PCA
sklearn.decomposition. KernelPCA



```
time_file = io.open("time_opt.txt", "w")
def doClusters(num_clusters, reducer, X, opt_file, i):
    start = time.time()
    if(reducer == 'pca'):
        pca = PCA(n_components=i)
        X = pca.fit_transform(X)
    elif(reducer == 'kpca,lin'):
        kpcal = KernelPCA(n_components=i,kernel='linear')
        X = kpcal.fit_transform(X)
    elif(reducer == 'kpca,poly'):
        kpcap = KernelPCA(n_components=i,kernel='poly')
        X = kpcap.fit_transform(X)
    elif(reducer == 'kpca,cos'):
        kpcac = KernelPCA(n_components=i,kernel='cosine')
        X = kpcac.fit_transform(X)
    elif(reducer == 'kpca,sig'):
        kpcas = KernelPCA(n_components=i,kernel='sigmoid')
        X = kpcas.fit_transform(X)
    elif(reducer == 'svd'):
        tsvd = TruncatedSVD(n_components=i)
        tsvc = tsvd.fit_transform(X)
    elif(reducer == 'none' and i != 141):
        return;
    rt = time.time()-start
    start = time.time()
    km = KMeans(n_clusters=num_clusters, init='k-means++', n_init=20, random_state= 0)
    y=km.fit_predict(X)
    ct = time.time()-start
    time_file.write("reducer: "+reducer+": "+str(i)+" dims - reduce time:"+str(rt)+
        ", cluster time:"+str(ct)+", total time: "+str(rt+ct)+"\n");
    print("reducer: "+reducer+": "+str(i)+" dims - done")
    confusion=contingency_matrix(actuallabels,y)
    write2d(opt_file, reducer+"--"+str(i), confusion, actuallabels, y)
```

3.4.2 Clustering

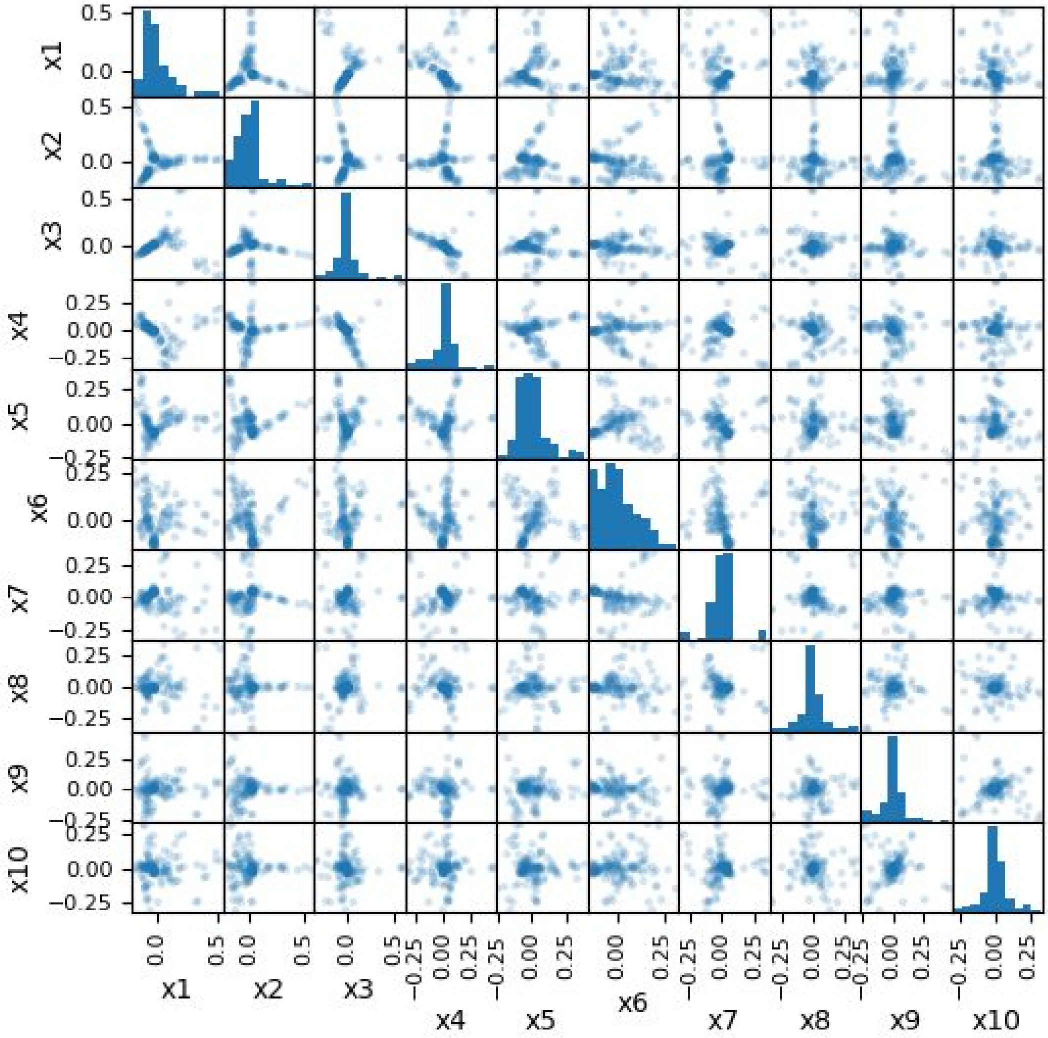
K-Means clustering used below is a representation of the clusters projected over 10 dimensions.



```
data = pd.DataFrame(X[:,:10], columns=['x1', 'x2', 'x3',
               'x4','x5','x6','x7','x8','x9','x10'])
k = scatter_matrix(data, alpha=0.2, figsize=(6, 6), diagonal='hist')
plt.show()
```

pandas.plotting.scatter_matrix

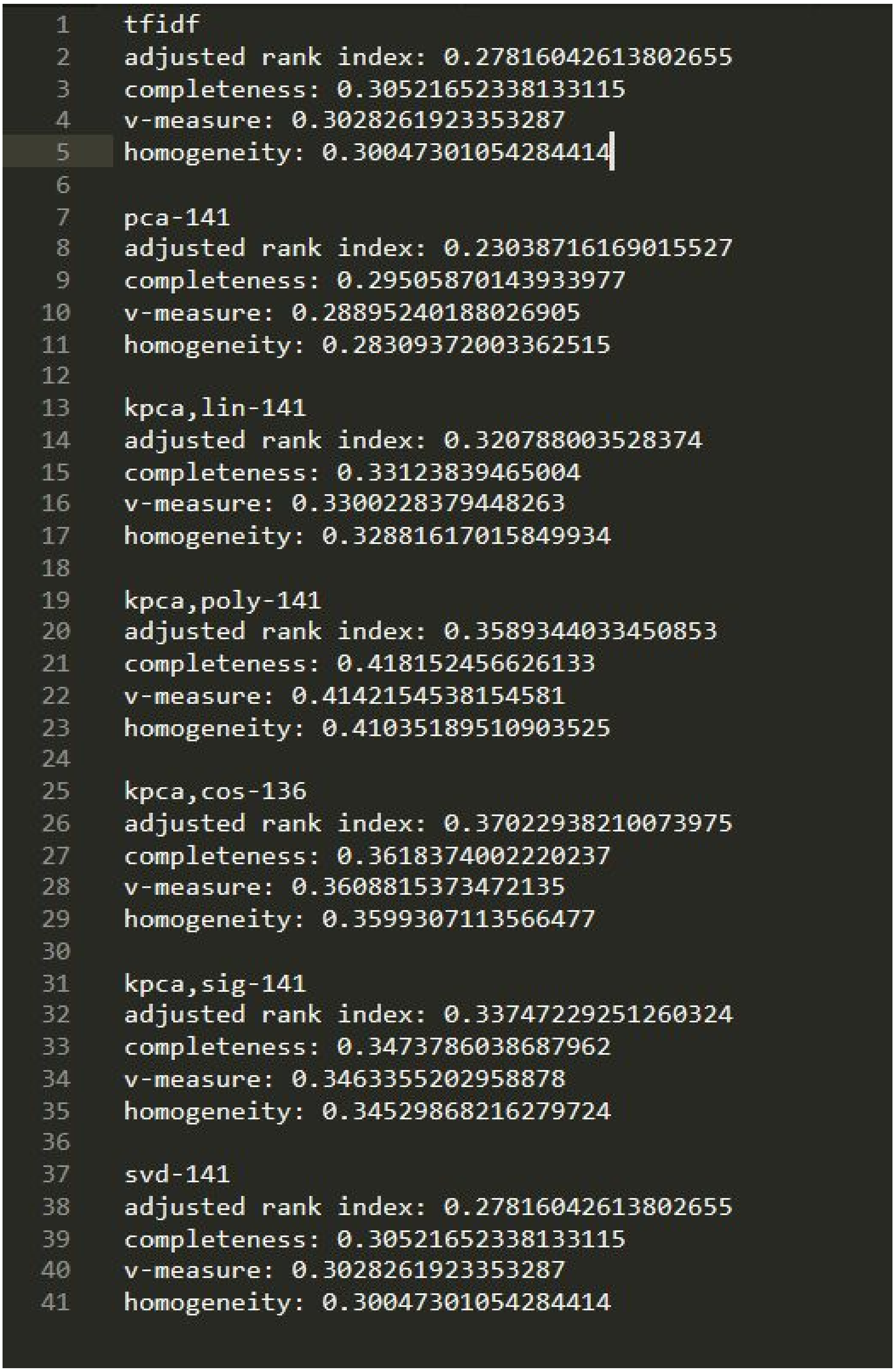
pandas.plotting.scatter_matrix(frame, alpha=0.5, figsize=None, ax=None, grid=False, diagonal='hist', marker='.', density_kwds=None, hist_kwds=None, range_padding=0.05, **kwds)



3.5 Cluster Evaluation

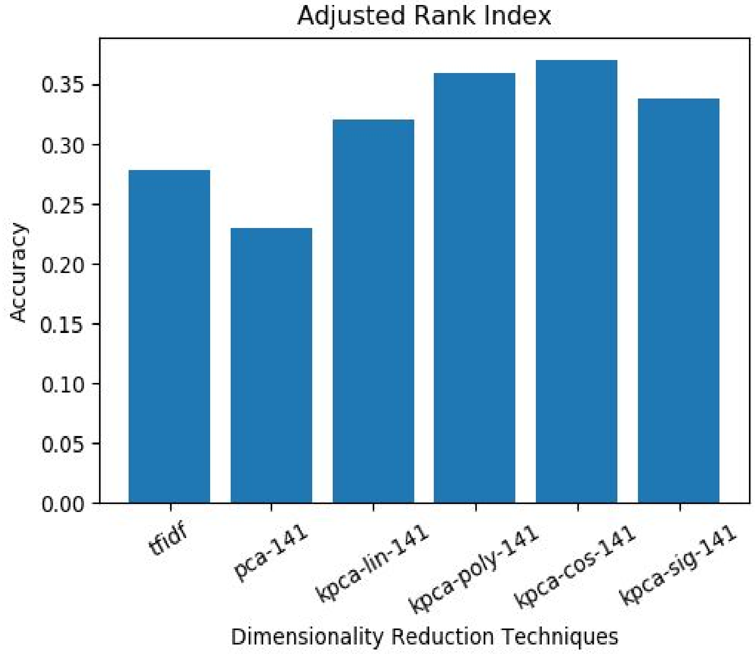
Techniques used are:

- Adjusted rand index
- Completeness
- Homogeneity
- V-measure

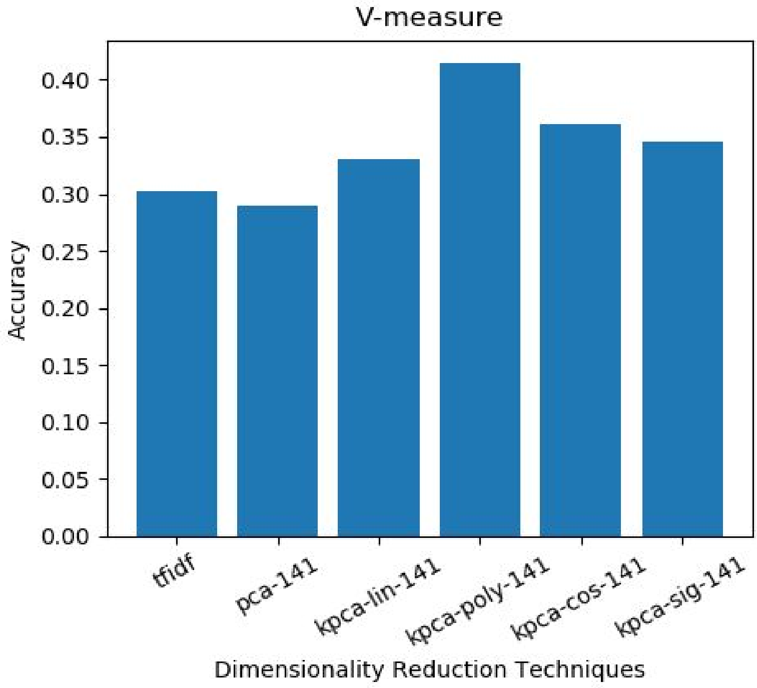


```
tfidf
adjusted rank index: 0.27816042613802655
completeness: 0.30521652338133115
v-measure: 0.3028261923353287
homogeneity: 0.30047301054284414

pca-141
adjusted rank index: 0.23038716169015527
completeness: 0.29505870143933977
v-measure: 0.28895240188026905
homogeneity: 0.28309372003362515

kpca,lin-141
adjusted rank index: 0.320788003528374
completeness: 0.33123839465004
v-measure: 0.3300228379448263
homogeneity: 0.32881617015849934

kpca,poly-141
adjusted rank index: 0.3589344033450853
completeness: 0.418152456626133
v-measure: 0.4142154538154581
homogeneity: 0.41035189510903525

kpca,cos-136
adjusted rank index: 0.37022938210073975
completeness: 0.3618374002220237
v-measure: 0.3608815373472135
homogeneity: 0.3599307113566477

kpca,sig-141
adjusted rank index: 0.33747229251260324
completeness: 0.3473786038687962
v-measure: 0.3463355202958878
homogeneity: 0.34529868216279724

svd-141
adjusted rank index: 0.27816042613802655
completeness: 0.30521652338133115
v-measure: 0.3028261923353287
homogeneity: 0.30047301054284414
```

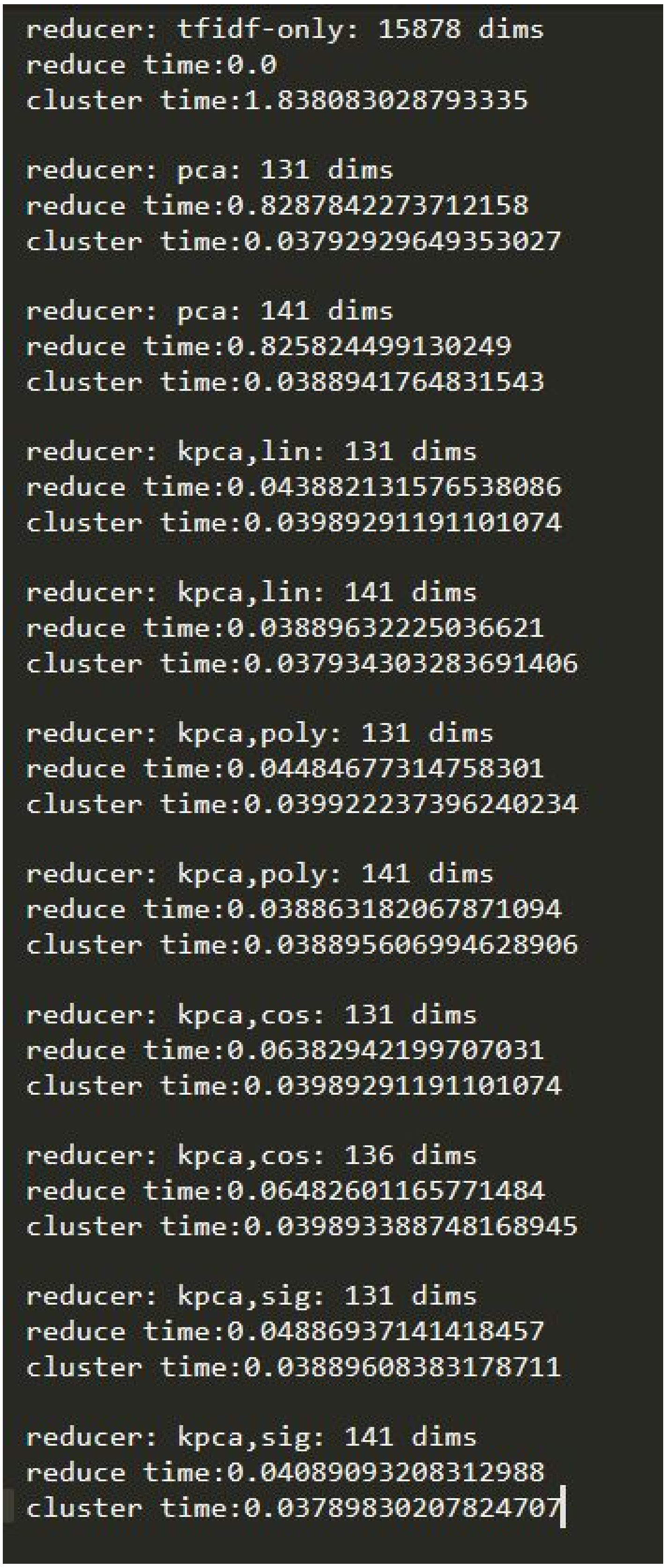
Adjusted rank index:



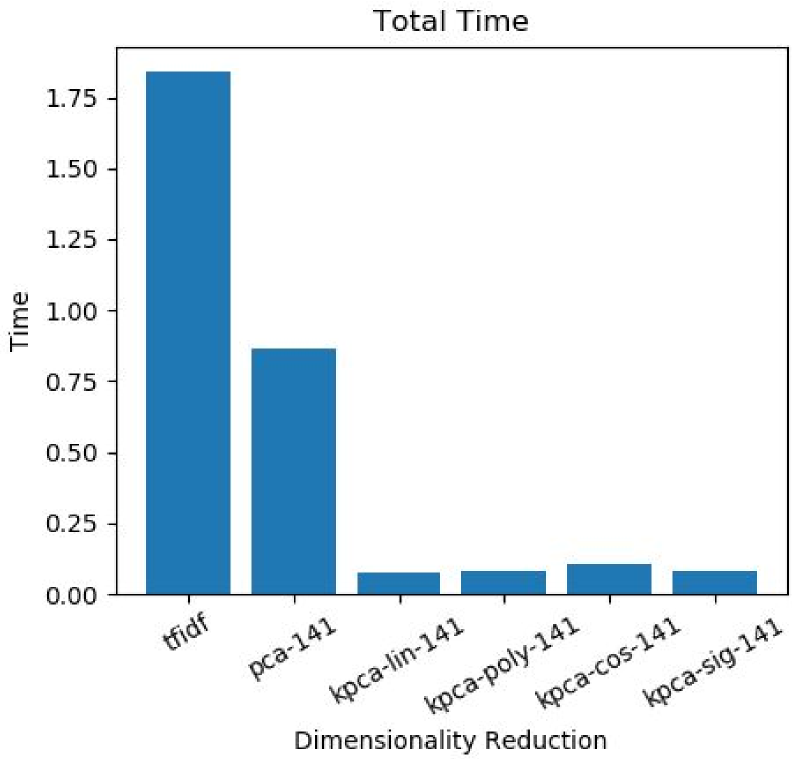
V-measure:



Clustering time:



```
reducer: tfidf-only: 15878 dims
reduce time:0.0
cluster time:1.838083028793335

reducer: pca: 131 dims
reduce time:0.8287842273712158
cluster time:0.03792929649353027

reducer: pca: 141 dims
reduce time:0.825824499130249
cluster time:0.0388941764831543

reducer: kpca,lin: 131 dims
reduce time:0.043882131576538086
cluster time:0.03989291191101074

reducer: kpca,lin: 141 dims
reduce time:0.03889632225036621
cluster time:0.037934303283691406

reducer: kpca,poly: 131 dims
reduce time:0.04484677314758301
cluster time:0.039922237396240234

reducer: kpca,poly: 141 dims
reduce time:0.038863182067871094
cluster time:0.038895606994628906

reducer: kpca,cos: 131 dims
reduce time:0.06382942199707031
cluster time:0.03989291191101074

reducer: kpca,cos: 136 dims
reduce time:0.06482601165771484
cluster time:0.039893388748168945

reducer: kpca,sig: 131 dims
reduce time:0.04886937141418457
cluster time:0.03889608383178711

reducer: kpca,sig: 141 dims
reduce time:0.04089093208312988
cluster time:0.03789830207824707
```


Total Time
Time
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
tfidf
pca-141
kpca-lin-141
kpca-poly-141
kpca-cos-141
kpca-sig-141
Dimensionality Reduction

4. CONCLUSION AND FUTURE SCOPE

The goal of text document clustering is to minimize the intra cluster distance between documents while maximizing the inter cluster distance using an appropriate distance measure between documents. With the full data representation, the K-Means algorithm cannot effectively clusters the document collection. By applying the Feature Selection/dimensionality reduction method with K-Means algorithm improves the clustering quality quite well. It has also been observed that while running the test data set on a machine with Intel core i5-8300H CPU @ 2.3GHz with 8GB RAM and 64-bit operating system K-Means with dimensionality reduction takes lesser time than K-Means with the full data representation to produce similar results.

In future, better stopword removal techniques, stemming and morphological analysis can be used to produce better clustering results.

5. REFERENCES

- https://medium.com/@dmitriy.kavyazin/principal-component-analysis-and-k-means-clustering-to-visualize-a-high-dimensional-dataset-577b2a7a5fe2
- https://www.kaggle.com/arthurtok/principal-component-analysis-with-kmeans-visuals
- https://towardsdatascience.com/dimensionality-reduction-for-machine-learning-80a46c2ebb7e
- https://en.m.wikipedia.org/wiki/Principal_component_analysis
- https://en.wikipedia.org/wiki/Kernel_principal_component_analysis
- https://insidebigdata.com/2018/07/26/what-is-text-clustering/
- http://webbut.unitbv.ro/
- https://scikit-learn.org/stable/modules/clustering.html#k-means
- https://scikit-learn.org/stable/modules/clustering.html#clustering-performance-evaluation