Sentiment Analysis

Sneha Mandan
DA-IICT, Gandhinagar
201401422@daiict.ac.in
Supervisor
Prof. Suman Mitra

***Abstract*—Sentiment analysis is the task of determining the reaction of public on certain topic. In this work, specific focus is made on this classification problem using dimensionality reduction techniques. Singular Value Decomposition (SVD) method and Locality Preserving Projections (LPP) with different metrics to find neighbours and weight functions have been implemented and performance is compared.**

I. Introduction

Sentiment analysis, or opinion mining is one of the important problem domain in natural language processing. Sentiment analysis is basically determining the reaction of public towards certain topic, phenomenon or product. The reviews made by people in blog posts, social medias or reviews on e-commerce websites can be analyzed. This task has utility in various domains such as in developing strategy for product launch in marketing, in making policy decisions, etc. [1]

Sentiment categorization is essentially a classification problem, where features that contain opinions or sentiment information should be identified before the classification. Sentiment analysis is often conducted at one of the three levels: the document level, sentence level, or attribute level. In this work, focus is on document level sentiment categorization.

II. Related Work

In the survey [2], the problem domain, potential applications, challenges and features are highlighted. The potential users can be an individual customer or company or government, across variety of domains. The challenges include identifying the subjective content in the document, finding the overall sentiment of the document or individual entity.

In [3], process for sentiment polarity categorization was proposed based on sentiment tokens (tokens that have sentiments based on dictionary of positive and negative tokens) and Part-of-speech tagging. Experiments were performed on Amazon.com reviews and NB, SVM and Random forest classification algorithm.

In [4], unsupervised probabilistic model was proposed to capture semantic term–document information from feature vectors (bag-of-words). It introduces a benchmark IMDb movie review dataset which is used in this work.

In [5], a classifier based on multinomial NB classifier that uses n-gram and POS-tags as features was proposed for Twitter dataset.

A lexicon-based approach to classify domain specific reviews was proposed in [6]. For each hotel review, to identify lexicons, word tokens, POS tag and sentiment score were used.

III. Proposed model

Broadly, there are four major steps (i) Preprocessing (ii) Feature Extraction (iii) Feature Selection (iv) Training and Testing

A. Preprocessing

The foremost step is to obtain clean data from the raw data. Here, the data is primarily text data in form of reviews in English language. Each review is termed as *document*. As a part of the preprocessing of *document*, the text is converted to lower case, punctuation and extra white spaces are removed and clean text is obtained. The stop words, that is common words such as ”is”, ”the”, ”and”, etc which do not contribute in the classification of document are eliminated from the text. Then, each document is tokenized based on *white space* parameter.

In addition to above, experiments have been done which include lemmatization or POS-tagging. Lemmatization refers to transforming inflected form of a word to its root form. For example, ”years” to ”year”, ”women” to ”woman” and so on. POS-tagging refers to assigning part-of-speech tag to each token. The POS-tagger is based on *Penn treebank project*. For our analysis, the tokens which have tags *adjective, adverb* or *verb* only are included as they contribute towards the sentiment in a sentence. [3]

B. Feature Extraction

The feature vector corresponding to each review/document is the corresponding TF-IDF (Term Frequency, Inverse Document Frequency) vector. *tf-idf score* are assigned based on the importance of tokens(or ”terms”) in a document depending on how frequently they appear across multiple documents. Intuitively, if a word appears frequently in a document, it is important attribute for classification but if a word appears in many documents, it’s not a unique identifier. Term frequency *tf*, inverse document frequency *idf* and term frequency-inverse document frequency *tfidf* for each token(term) are obtained as follows:

$$tf = \frac{f}{tt} \tag{1}$$

$$idf = ln\frac{N+1}{df+1} + 1 \tag{2}$$

$$tfidf = tf * idf \tag{3}$$

where f is frequency of each term in a document, *tt* is total number of terms in a document, N is total number of documents and *df* is frequency of a term across all documents.

Thus, we obtain *tf-idf matrix* where rows correspond to documents and columns correspond to terms. One of the important points to note here is that the *tf-idf matrix* obtained is a sparse matrix. The *tfidf vectors* act as feature for our machine learning model.

Experiments have been done while retaining only the features (terms) which have document frequency above certain threshold (1%).

C. Feature Selection

The number of terms in a text corpora is huge. Hence the feature space has large dimension in comparison to number of input samples. Also, not all features contribute in classifying the documents. Our main aim is to reduce the dimensions of the feature space retaining the important information content. The feature vector is projected onto space with lower dimensions, which acts as input for the machine learning model. If there are *n* documents and *d* terms, the dimension of *tf-idf matrix* is *n* x *d*. We want to obtain projected matrix with *n* x *k* dimensions, where *k* << *d*.

The techniques used for this purpose are Singular Value Decomposistion (SVD) and Locality Preserving Projection (LPP) and its variants.

D. Dimensionality Reduction

1) Singular Value Decomposition: SVD is a method for identifying and ordering the dimensions along which data points exhibit the most variation. The singular value decomposition of a matrix A is the factorization of A into the product of three matrices $A = UDV^T$ where the columns of U and V are orthonormal and the matrix D is diagonal with entries corresponding to eigenvalues in decreasing order [7]. The projection in lower space is obtained as $A_k = UD_k^T$ where only k-significant eigenvectors are considered.

2) Locality Preserving Projection: This linear transformation optimally preserves local neighborhood information [8]. The algorithm is as follows:
Step 1 Neighbourhood matrix: Construct neighbourhood matrix S_{nxn} for documents. The value $S[i,j]$ corresponds to distance metric for documents i & j if they are neighbours else zero.
Step 2 Weight matrix: Construct weight matrix W_{nxn} for documents. The value $W[i,j]$ is the weight score assigned to neighbouring documents based on some function of distance between them.
Step 3 Eigen value eqaution: Solve generalized eigen value problem $XLXTv = \lambda XDX^Tv$. Here, D is diagonal matrix with entries equal to column sum of W and $L = W - D$.
Step 4 Projection matrix: Project original feature vector in space with reduced dimension corresponding to first k eigen vectors.

Here, the eigen-values obtained in increasing order correspond to direction that minimizes the local variance between the points(documents) and hence preserving the locality.

In supervised LPP, we leverage the knowledge of class labels. Thus, look for neighbouring documents only among the documents of same class label. As a result, the neighbourhood matrix S will be a block-diagonal matrix.

The standard distance metric to find neighbours is euclidean distance and weight is assigned based on exponential function [8]. In addition to these, different distance metrics to find neighbours and functions to assign weight score were experimented with and performance was compared. Experiments have been performed wherein (i) neighbours are found based on cosine similarity and weights score are assigned as cosine similarity. (ii) neighbours are found based on Mahalanobis distance and weights score are assigned based on z-shaped function. Experiments have been performed with varying number of neighbours considered. Also, projection of feature space onto space with varying number of dimensions is implemented and performance is compared.

For two vectors u and v, the distance between them is defined as:
Euclidean distance $d(u,v) = \sqrt{\sum_{i=1}^{i=n}(u_i - v_i)^2}$
Cosine similarity $d(u,v) = \frac{u.v}{\sqrt{u.u}\sqrt{v.v}}$
Mahalanobis distance $d(u,,v) = \sqrt{(u-v)^T V^{-1}(u-v)}$ where V is the feature covariance matrix.

For distance x, the weight functions are defined as below.
Z-shaped function is defined as follows, where a and b are taken as minimum and maximum values respectively

$$zmf(x) = \begin{cases} 1 & x \leq a \\ 1 - 2(\frac{x-a}{b-a})^2 & a \leq x \leq \frac{a+b}{2} \\ 2(\frac{x-b}{b-a})^2 & \frac{a+b}{2} \leq x \leq b \\ 0 & x \geq b \end{cases}$$

Exponential function is defined as follows, where σ is the maximum variance

$$exp(x) = e^{-\frac{x}{\sigma^2}}$$

E. Training the model

The data is split into training set and test set in ratio 7:3 maintaining the proportion of samples of different classes in train and test set. *k-fold cross validation* with *k=5* is implemented for stable modeling of data set. *k-nearest Neighbours Classifier* is used for training. The neighbours are determined based on cosine metric. Let the input *tfidf* *matrix* be denoted as A_{Nxd} where rows represent documents (N documents) and columns represent features, i.e. terms (d *features*), the train set be denoted as X_{mxd} and test set as Y_{nxd}.

In implementation of SVD, from train data $X_{mxd} = U_{mxr} D_{rxr} V_{rxd}^T$ (where r=min{m,d}), projection in reduced space of k dimension ($k << d$) is obtained as $Xsvd_{mxk} = U_{mxk} S_{kxk}$. $Xsvd$ is used as input for machine learning model. For test data Y_{nxd}, projection into reduced space is obtained as $Ysvd_{nxk} = Y_{nxd} V_{dxk}$.

In LPP, to find neighbouring documents, before implementing distance metric, Principal component analysis (PCA) is implemented [9]. Hence, $Xpca_{mxm}$ is obtained as PCA projection of tf-idf matrix X_{mxd}. Then LPP algorithm is applied on $Xpca$. PCA is same as applying SVD on mean-centred data [10].

IV. Data Set and Tools

Data from various micro-blogging sites such as twitter, review sites, blogs can be used for sentiment analysis. In this project, the data sets used were of varied products/services. From the data set, review text and corresponding rating were extracted. The class label for each review was decided based on the rating.

The first dataset consists of benchmark movie review dataset [11]. It consists of 50k movie reviews , 25k positive and 25k negative reviews. The second dataset consists of hotel reviews [12] with rating as ”happy”(positive) or ”not happy”(negative). The third dataset consists of reviews from Amazon.com [13] comprised of product of single category, in this case video. The fourth dataset is of Amazon product reviews [14] consisting of different electronic products. These review ratings ranged from 1-5. The reviews with rating 1 or 2 were considered to be negative reviews, 4 or 5 as positive and rest as neutral.

Dataset	Positive	Negative	Neutral	Total
IMDb	1000	1000	-	2000
hotel reviews	1341	658	-	2000
amazon1	700	700	700	2100
amazon2	977	76	124	1177

Table I: Number of samples in each class for all datasets

All the preprocessing tasks are achieved using *nltk* package in *python*. *tfidf vectorizer* of *sklearn* package is used for feature extraction. *scipy* package was used to handle sparse matrices. *sklearn* package was used for implementing SVD, kNN classifier and performance metrics. *numpy* and *pandas* package were used to handle arrays and matrices.

A. Performance Analysis

1) Evaluation Metrics: The metrics used to determine performance of various algorithms are *accuracy* and *f1-score*. *True positives (TP)* and *True Negative (TN)* are correctly classified positive and negative samples, respectively. *False Positive (FP)* and *False Negative (FN)* denotes misclassified positive and negative samples, respectively.
Accuracy = (TP + TN)/(TP + TN + FP +FN)
Recall = TP/(TP + FN)
Precision = TP/(TP+FP)
f1-score = 2(Recall)(Precision)/(Recall + Precision)

Table-II highlights the average accuracy for IMDb dataset corresponding to different preprocessing techniques. *distance* and *weight* denote the distance metric and weight function, respectively, used in LPP. n denotes the number of nearest neighbours considered in LPP. *dim* denotes the number of dimensions (or the eigen values) used to project the data into reduced dimensional space.

	SVD	LPP	LPP	SLPP	LPP	SLPP
distance	-	euc	cos	cos	md	md
weight	-	exp	cos	cos	zmf	zmf
P1	**86.57**	**58.6**	**85.23**	**86.1**	**86.37**	**83.57**
dim	1000	1000	1000	500	1000	10
neighbours	-	100	100	90	5	5
P2	80.9	53.17	80.47	80.63	81.13	78.97
dim	500	1000	1000	1000	1000	1000
neighbours	-	5	50	100	90	100
P3	86.1	62.2	84.77	85.63	85.97	84.77
dim	1000	1000	1000	500	500	500
neighbours	-	100	50	25	100	5
P4	81.97	59.2	81.07	80.77	82.13	77.43
dim	1000	1000	1000	1000	1000	1000
neighbours	-	100	90	90	50	15

Table II: Average accuracy (in %) for preprocessing techniques on IMDb dataset. P1 is without lemmatization or POS-tagging (features=22919). P2 is without lemmatization or POS-tagging and $document frequency \geq 1\%$ (features=1627). P3 is with lemmatization (features=20650). P4 is with lemmatization and POS-tagging (features=12733).

Based on above results, the preprocessing without lemmatization or POS-tagging yields the best results for given datasets. Average and maximum *accuracy* and *f1-score* are reported.

Table-III highlights the classification accuracy on all datasets without any dimensionality reduction algorithm, in the original feature space. Hence, original *tf-idf vector* is used as input for classifier algorithm. The preprocessing step doesn't involve lemmatization or POS-tagging.

	IMDb	hotel	amazon1	amazon2
dim	22919	10489	16586	5242
avg acc(%)	86.4±1.2	79.6±1.3	58.57±1	86.67±0.7
avg f1(%)	86.38±1.22	76.91±1.7	58.27±1.2	83.32±1.2
max acc(%)	87.5	80.83	59.84	87.57
max f1(%)	87.49	78.6	58.85	85

Table III: Performance of all datasets without dimensionality reduction

The performance of various dimensionality reduction algorithms is compared in following tables (tables - IV to XI).

	distance	weight	avg acc(%)	avg f1(%)	dim	n
SVD	-	-	**86.57**±**2**	86.5±2	1000	-
LPP	euc	exp	58.6±1.5	58.54±1.5	1000	100
LPP	cos	cos	85.23±1.4	85.21±1.4	1000	100
SLPP	cos	cos	86.10±1.6	86.07±1.6	500	90
LPP	md	zmf	**86.37**±**1.6**	86.34±1.6	1000	5
SLPP	md	zmf	83.57±3	83.52±3	10	5

Table IV: IMDb dataset average performance (number of features is 22919). *dim*-number of reduced dimensions. LPP parameters - *distance metric*, *weight function*, *n*-number of nearest neighbours.

	distance	weight	acc(%)	f1(%)	dim	n
SVD	-	-	88.5	88.49	1000	-
LPP	euc	exp	61	60.9	1000	100
LPP	cos	cos	87.5	87.48	1000	100
SLPP	cos	cos	88.33	88.33	500	75
LPP	md	zmf	88.33	88.3	1000	5
SLPP	md	zmf	**88.67**	88.64	100	5

Table V: IMDb dataset best performance (number of features is 22919). *dim*-number of reduced dimensions. LPP parameters - *distance metric*, *weight function*, *n*-number of nearest neighbours.

	distance	weight	avg acc(%)	avg f1(%)	dim	n
SVD	-	-	81.43±1.3	80.24±2.7	250	-
LPP	euc	exp	67.27±1.4	59.84±1.7	1000	100
LPP	cos	cos	80.97±1.3	79.5±1.5	1000	5
SLPP	cos	cos	80.67±1.3	78.94±1.6	1000	5
LPP	md	zmf	**82.8**±**1.3**	81.72±1.5	1000	75
SLPP	md	zmf	78.61±1.3	77.28±1.3	1000	100

Table VI: hotel-reviews average performance (number of features is 10489). *dim*-number of reduced dimensions. LPP parameters - *distance metric*, *weight function*, *n*-number of nearest neighbours.

	distance	weight	acc(%)	f1(%)	dim	n
SVD	-	-	83.17	81.71	1000	-
LPP	euc	exp	69	62.12	1000	100
LPP	cos	cos	82	80.62	1000	5
SLPP	cos	cos	82	80.57	1000	5
LPP	md	zmf	**84.83**	83.9	1000	75
SLPP	md	zmf	81.17	80.86	500	75

Table VII: hotel-reviews dataset best performance (number of features is 10489). *dim*-number of reduced dimensions. LPP parameters - *distance metric*, *weight function*, *n*-number of nearest neighbours.

	distance	weight	avg acc(%)	avg f1(%)	dim	n
SVD	-	-	58.92±0.7	58.35±0.6	1000	-
LPP	euc	exp	40.09±1.9	40.02±1.8	1000	100
LPP	cos	cos	58.32±1.3	57.03±1.6	1000	100
SLPP	cos	cos	58.06±0.7	56.94±0.7	1000	5
LPP	md	zmf	**59.62**±**1.3**	58.71±1.5	1000	5
SLPP	md	zmf	58.09±1.9	58.13±1.8	1000	100

Table VIII: Amazon dataset1 average performance (number of features is 16586). *dim*-number of reduced dimensions. LPP parameters - *distance metric*, *weight function*, *n*-number of nearest neighbours.

	distance	weight	acc(%)	f1(%)	dim	n
SVD	-	-	60.95	60.78	5	-
LPP	euc	exp	43.01	43.04	1000	100
LPP	cos	cos	60	58.79	1000	100
SLPP	cos	cos	60.47	59.61	1000	100
LPP	md	zmf	**61.27**	60.54	1000	5
SLPP	md	zmf	60.79	60.57	1000	100

Table IX: Amazon dataset1 best performance (number of features is 16586). *dim*-number of reduced dimensions. LPP parameters - *distance metric*, *weight function*, *n*-number of nearest neighbours.

	distance	weight	avg acc(%)	avg f1(%)	dim	n
SVD	-	-	87±0.8	83.85±1.5	100	-
LPP	euc	exp	86.27±0.3	81.91±0.6	500	50
LPP	cos	cos	86.55±0.7	82.71±1.1	500	50
SLPP	cos	cos	**88.24**±**0.9**	86.22±1.6	250	50
LPP	md	zmf	86.95±0.8	83.64±1.3	500	25
SLPP	md	zmf	87.23±0.8	84.43±1.3	500	25

Table X: Amazon2 dataset average performance (number of features is 5242). *dim*-number of reduced dimensions. LPP parameters - *distance metric*, *weight function*, *n*-number of nearest neighbours.

	distance	weight	acc(%)	f1(%)	dim	n
SVD	-	-	88.42	86.16	250	-
LPP	euc	exp	86.72	82.67	5	5
LPP	cos	cos	87.85	84.47	500	50
SLPP	cos	cos	**89.27**	88.8	250	50
LPP	md	zmf	88.14	84.86	250	50
SLPP	md	zmf	88.98	87.38	250	25

Table XI: Amazon2 dataset best performance (number of features is 5242). *dim*-number of reduced dimensions. LPP parameters - *distance metric*, *weight function*, *n*-number of nearest neighbours.

V. Conclusion

Amongst various preprocessing techniques, the one without lemmatization or POS-tagging is shown to give higher classification accuracy. In preprocessing, if the features with document-frequency below 1% are excluded, the dimension of original feature space reduces drastically (For IMDb dataset, by 92%) and giving acceptable accuracy (81% in comparison to 86.57%). In comparison to original feature space, better performance is achieved by retaining only 4%-10% features in reduced dimension space. Variants of LPP with cosine similarity and mahalanobis distance to find neighbours perform better than standard euclidean distance based neighbourhood. SLPP with cosine similarity is better than its unsupervised variant as similar or better performance is achieved while retaining lesser number of features and considering smaller neighbourhood. SVD for IMDb dataset, LPP with mahalonobis distance metric for hotel review dataset and amazon1 dataset and SLPP with cosine similarity give best performance for amazon2 dataset.

If computational resources are available, training the model on huge data-set may yield better results. In addition to document level analysis, entity level and sentence level sentiment categorization can be implemented.

References

[1] G Vinodhini and RM Chandrasekaran. Sentiment analysis and opinion mining: a survey. *International Journal*, 2(6):282–292, 2012.

[2] Bo Pang, Lillian Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135, 2008.

[3] Xing Fang and Justin Zhan. Sentiment analysis using product review data. *Journal of Big Data*, 2(1):5, 2015.

[4] Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 142–150. Association for Computational Linguistics, 2011.

[5] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10, 2010.

[6] Dietmar Gräbner, Markus Zanker, Günther Fliedl, and Matthias Fuchs. Classification of customer reviews based on sentiment analysis. 05 2012.

[7] Kirk Baker. Singular value decomposition tutorial. *The Ohio State University*, 24, 2005.

[8] Xiaofei He and Partha Niyogi. Locality preserving projections. In *Advances in neural information processing systems*, pages 153–160, 2004.

[9] Zizhu Fan, Ming Ni, Meibo Sheng, Zejiu Wu, and Baogen Xu. Principal component analysis integrating mahalanobis distance for face recognition. In *Robot, Vision and Signal Processing (RVSP), 2013 Second International Conference on*, pages 89–92. IEEE, 2013.

[10] Jonathon Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.

[11] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[12] Hotel-reviews-data-kaggle. https://www.kaggle.com/harmanpreet93/hotelreviews.

[13] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52. ACM, 2015.

[14] Amazon2-data. https://data.world/datafiniti/consumer-reviews-of-amazon-products.