Sentiment Analysis: SVM and Kernel Trick

Bichu George, *CB.EN.P2DSC21008*, I^{st} *year MTech* – *DataScience*,
Batch(2021-2023), Course: 21DS601 and 21MA602(21-22(Odd)),
e-mail: cb.en.p2dsc21008@cb.students.amrita.edu

Abstract

Sentiment analysis (also known as opinion mining or emotion AI) is the use of natural language processing, text analysis, computational linguistics, and bio-metrics to systematically identify, extract, quantify, and study effective states and subjective information. Sentiment analysis is widely applied to voice of the customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine. Several Tweets are collected for our work from twitter. These tweets can be gathered and the sentiment can be identified using Machine Learning and Natural Language Processing(NLP) techniques. In this work, we used different SVM Kernel functions to classify tweets into positive or negative.

Index Terms

Machine Learning, SVM, Kernel Trick, t-SNE, PCA, Natural Language Processing, Metric Evaluation.

I. Introduction

NATURAL Language Processing is a brach of computer science- specifically, branch of artificial intelligence by which computers acquire the ability to interpret textual data and speech in much the same way humans can. We know that, compared to statistical or numerical data, textual data has a higher dominance. Therefore, NLP can be applied to variety of practical applications by making use of the massive raw data. Some famous applications of NLP consists of statistical machine translations [1] used by Google Translate, siri, Google Assistant etc... Apart from these 'regular use' categories, there are some sensitive applications as well. That include fraud detection systems, detecting fake news, profiles on social media.[2]. Due to many advancements in applied mathematics and computer science, more opportunities have opened towards the sophisticated machine learning algorithms.

MACHINE Learning is used to detect cancer[3]. Most of the data used here are either image or statistical data. Nevertheless, some amount of uncertainty will occur in our mind to trust a computer for such sensitive tasks. In this work, we try to classify the medical transcriptions and apply various machine learning algorithms to perform a comparative analysis in detecting medical speciality. Class imbalance is a natural thing in real life. We also consider this issue in our work. In Section IV we visualize-data using t-SNE. In Section V we preprocess the textual data and handles class imbalance. Very brief description of algorithms used is mentioned in section IV. Later sections deals with the performance of our algorithms on two different Kernels that we have used in our study.

Dr. Soman K. P,
Head Of the Department
Computational Engineering and Networking, Amrita School of Engineering,
Amrita Vishwa Vidyapeetham, Coimbature, India

II. Literature Review

Support vector machine (SVMs) algorithms have strong theoretical foundations and excellent empirical successes. They are seen to be more interpret-able than deep neural networks, and it had been mostly used in many classification problems such as handwritten digit recognition, object recognition, and text classification. The support vector machines are widely used due to their relatively powerful performance over many different areas. The authors in [] explore two main problems: text analysis in natural language processing and finding the best algorithm in machine learning. Several of the most wellknown classification methods are considered: the Bayesian method, the Support Vector Machines method (SVM), and the K nearest neighbor method. Experiments on the classification of a collection of texts in English, Chinese and Russian prove that SVM has an advantage over other methods in accuracy and completeness. Analysis of texts in Russian and English on the basis of machine learning methods is presented in [15] The aim of the study is to test and compare different methods of machine learning, of which: Bayesian method, K nearest neighbor method, Rocchio method, SVM, key classification method words and its combination with SVM. In all approaches, the weights of the words in the text are determined by the TF.IDF scheme and English and Russian "stop words" are excluded. The metrics used for comparisons are accuracy, precision, completeness and F-measure. Studies show that SVM and the combined method give the best results.

Classification of medical documents in Georgian language into three groups, mainly X-Ray, endoscopy and ultrasonography is performed by Khachidze et. al.in[6]. Khachidze achieved this with the help of four basic task of NLP, ie, Tokenization, Lemmatization, text-preprocessing and classifiers. Comparing Support vector machines(SVM) and K-nearest neighbhour(K-NN), it was found that SVM had a better accuracy than K-NN. . A unique statistical text classifier was developed by Ong et. al.[7] in which detects extreme risks. Classifiers based on Naive Bayes and SVM was utilized to achieve this task and it was trained as well as tested on clinical incident reports. Classification took place by mapping the features to a higher dimensional space with the help of some kernel functions like: linear, polynomial and RBF in SVM. Linear SVM showed a higher performance commpared to other kernel functions and Naive Bayes performed reasonably. An empirical study of three text classification algorithms is presented by the authors in [] , using two data sets. The Naive Bayes Classifier, the Support Vector Machine Method, and the Solution Tree (C4.5) are studied by training instances of the Weka tool datasets. The results are compared based on the values of precision and sensitivity for each of the algorithms. Studies show that SVM is the most effective among the three classifiers.

III. Objective

The main objective of this work is to to a comparative analysis of SVM Kernels to correctly classify the sentiments from tweets.

IV. Theoretical Background

Our approach make use of a classical machine learning algorithm known as Support Vector Machines (SVM) and its Kernels such as Linear and Gaussian. A brief algorithm literature for each of the algorithms are described as follows:

A. *Support Vector Machine*

The first paper published by Vapnik, Chervonenkis and their coworkers [10] went unnoticed till 1992. Classification in SVM is done by constructing a hyperplane which will separate two classes.

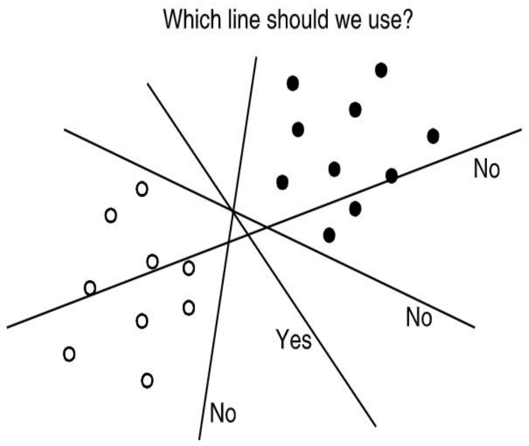


Fig. 1. Choosing the best plane for classification.

Separating the classes by a large margin will result in reducing the expected generalization error. Which means that when an unseen test data comes, the chance of miss-classification is low. Support vectors are defined as the planes which are parallel to the separating hyperplane and which pass through any one or more points in the data-set.

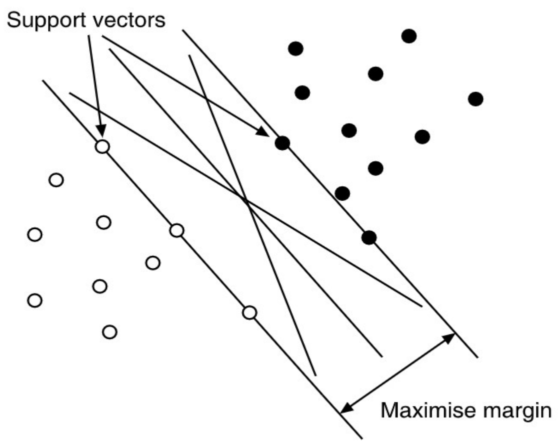


Fig. 2. Closer look on Support vectors.

The points in the data-set which will fall on the bounding plane are called support vectors. There may be scenarios that the data-points are closely clustered so that we cannot linearly separate the two classes. in this case we prefer a non-linear map of data to some higer dimensional space, which we call "feature space", there it will be linearly separable. Hence we call it a non-linear classifier.

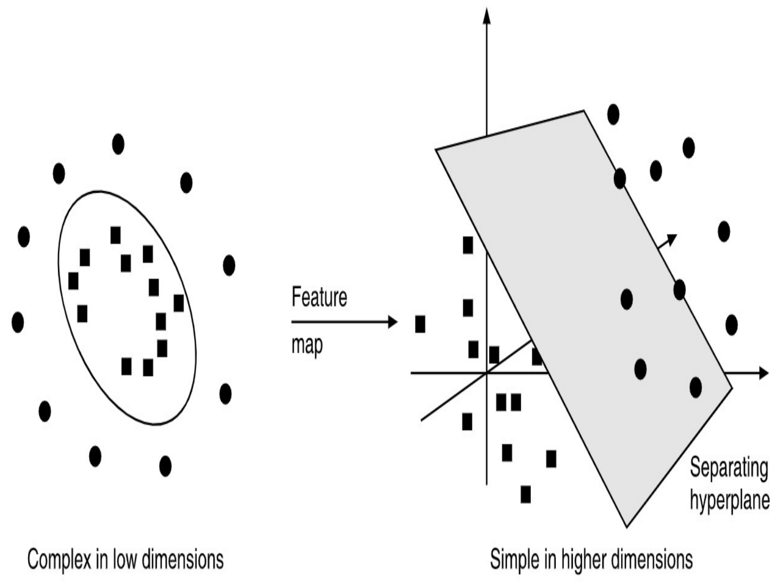


Fig. 3. Mapping to feature space.

If we consider the two-class linearly separable data, (x_i, y_i) for i=1,2,..,m and class values $y_i \in (-1,1)$ where number of data points are m and it is in n dimensional space. The separating hyperplane is defined as, $w^T x - \gamma = 0$, $\quad w \in R^n$. So the decision function is $f(x) = sign(w^T x - \gamma)$. The primal form of the two class problem for hard margin is,

$$\begin{aligned} \min_{w,\gamma} \quad & \frac{1}{2} w^t w \\ \text{s.t.} \quad & D(Aw - \gamma e) \geq e \end{aligned} \tag{1}$$

The primal form of the two class problem for soft margin is,

$$\begin{aligned} \min_{w,\gamma,\xi} \quad & \frac{1}{2} w^t w + C \sum_{i=1}^{N} \xi_i \\ \text{s.t.} \quad & y_i(w^T \phi(x_i) - \gamma) + \xi_i - 1 \geq 0 \\ & \xi \geq 0 \end{aligned} \tag{2}$$

Where $K(x_i, x_j) = \langle \phi(x_i) \cdot \phi(x_j) \rangle$ is the kernel function and $\phi \colon R^n \to R^k$ is a mapping function used to project data to higher dimension.

1) ***Linear Classifier assuming complete separability:*** Let m = number of points in the data-set and n= number of features in the data. $x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ . \\ . \\ . \\ x_{in} \end{bmatrix}$ $d_i = D_{ii} =$ Target value of ith data. $d = \begin{bmatrix} d_1 \\ d_2 \\ . \\ . \\ . \\ d_m \end{bmatrix}$ Vector representing target value of n data points.

$$D = diag(d) = \begin{bmatrix} d_1 & 0 & 0 & ... & 0 \\ 0 & d_2 & 0 & ... & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ 0 & 0 & ... & 0 & d_m \end{bmatrix} \qquad w = \begin{bmatrix} w_1 \\ w_2 \\ . \\ . \\ . \\ w_n \end{bmatrix}$$ Coefficient vector corresponding to hyper-plane.

$w_1x_1 + w_2x_2 + ... + w_nx_n - \gamma = 0$. γ is scalar which is generally known as bias term.

$$A = \begin{bmatrix} x_1^T \\ x_2^T \\ . \\ . \\ . \\ x_m^T \end{bmatrix} \qquad AA^T = \begin{bmatrix} x_1^Tx_1 & x_1^Tx_2 & ... & x_1^Tx_m \\ x_2^Tx_1 & x_2^Tx_2 & ... & x_2^Tx_m \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ x_m^Tx_1 & x_m^Tx_2 & ... & x_m^Tx_m \end{bmatrix}$$ AA^T is called the linear kernel of the data-set.

$\phi(.) \rightarrow x$ is the non-linear mapping function that maps input vector x into a higher dimensional feature space.

$$K = \begin{bmatrix} \phi(x_1)^T\phi(x_1) & \phi(x_1)^T\phi(x_2) & ... & \phi(x_1)^T\phi(x_m) \\ \phi(x_2)^T\phi(x_1) & \phi(x_2)^T\phi(x_2) & ... & \phi(x_2)^T\phi(x_m) \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ \phi(x_m)^T\phi(x_1) & \phi(x_m)^T\phi(x_2) & ... & \phi(x_m)^T\phi(x_m) \end{bmatrix}$$ K is called the non-linear kernel of the input data-set.

Q an mxm matrix whose $(i,j)th$ element is $d_id_j\phi(x_i)^T\phi(x_j)$.
$Q = K.*(d*d^T)$ where $.*$ represents element wise multiplication.
$svindx$ denotes the index of support vectors.

*2) **Lagrangian Dual: Matrix form**:* In the matrix form, the primal problem is

$$\begin{aligned} \min_{w,\gamma} \quad & \frac{1}{2}(w^Tw) \\ \text{Subject to,} \quad & D(AW - e\gamma) \geq e \end{aligned} \tag{3}$$

Where e is an m x 1 vector of ones. On applying Lagrangian on (3) we get,

$$L(w,\gamma,u) = \frac{1}{2}(w^Tw) - u^T((Aw - r\gamma) - e) \tag{4}$$

The solution is given by,

$$\max_{u} \min_{w,\gamma} L(w,\gamma,u) \tag{5}$$

Differentiating L wrt.. primal variables we get,

$$\frac{\delta L(w,\gamma,u)}{\delta\gamma} = 0 \implies e^T D^T u = e^T Du = 0 \tag{6}$$

$$\frac{\delta L(w,\gamma,u)}{\delta w} = 0 \implies w - A^T Du = 0 \tag{7}$$

Substituting $w = A^T Du$, we get,

$$L_D(u) = \frac{1}{2}(A^T Du)^T (A^T Du) - u^T (D(AA^T Du - e\gamma) - e) \tag{8}$$

$$L_D(u) = \frac{1}{2}(u^T DAA^T Du) - u^T DAA^T Du + \gamma u^T De + u^T e \tag{9}$$

Note that every term in the abouve expression is a scalar and the third term $\gamma u^T De$ is same as $\gamma e^T Du$. Since, $e^T Du = 0$,

$$L_D(u) = u^T e - \frac{1}{2} u^T DAA^T Du \tag{10}$$

So, the dual optimization problem is,

$$\begin{aligned} \max \quad & L_D(u) = u^T e - \frac{1}{2} u^T DAA^T Du \\ \text{Subject to,} \quad & e^T Du = 0, u \geq 0 \end{aligned} \tag{11}$$

However the standard form is,

$$\begin{aligned} \min \quad & L_D(u) = \frac{1}{2} u^T Qu - e^T u \\ \text{Subject to,} \quad & d^T u = 0, u \geq 0 \end{aligned} \tag{12}$$

Decision function $f(x)$ is given by,

$$f(x) = sign(w^T x - \gamma) \tag{13}$$

*3) **Compute γ using quadratic programming**:* For computing γ, one of those points whose Lagrangian multiplier falls between 0 and C can be used. That is, one of those points which falls on the hyper-planes. For these points, $w^T x_i - \gamma = d_i$. And therefore,

$$\gamma = w^T x_i - d_i = \sum_{j \in svindex} u_j d_j x_j^T x_i - d_i \tag{14}$$

It is customary to get γ from all such points and find average. Here svindex corresponds to the set of indices of support vectors including those points falling on the hyper-plane.

*4) **RBF Kernel:*** The radial basis function is defined by, $k(x,y) = exp(-\sigma||x-y||^2)$, where σ is a positive parameter controlling the radius. expanding $exp(-\sigma||x-y||^2)$, we obtain,

$$exp(-\sigma||x-y||^2) = exp(-\sigma||x||^2)exp(-\sigma||y||^2)exp(2\sigma x^T y) \tag{15}$$

since, $exp(2\sigma x^T y) = 1 + 2\sigma x^T y + \frac{(2\sigma)^2}{2!}(x^T y)^2 + \frac{(2\sigma)^3}{3!}(x^T y)^3 + ...$

$exp(2\sigma x^T y)$ is an infinite summation of polynomials. Thus, it is a kernel whose mapping function maps a point to an infinite dimensional space.

*B. **PCA***

Principal Component Analysis is a method that is used to reduce the dimension of a data-set. PCA is a linear transformation that takes a new coordinate system for the data-set such that strongest variance by any projection of the data-set comes to be on the first axis (this is called the first principal component) and the second largest variance on the second axis and so on. It is used to reduce dimensionality by eliminating the other principal components. Let $x_1, x_2, ..., x_n$ be a set of Nx1 vectors and let $\bar{x}$ be their mean.

$$x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \cdot \\ \cdot \\ \cdot \\ x_{iN} \end{bmatrix} \qquad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} \begin{bmatrix} x_{i1} \\ x_{i2} \\ \cdot \\ \cdot \\ \cdot \\ x_{iN} \end{bmatrix}$$

Let X be the Nxn matrix with columns $x_1 - \bar{x}, x_2 - \bar{x}, ..., x_n - \bar{x}$

$$X = \begin{bmatrix} x_1 - \bar{x} & x_2 - \bar{x} & ... & x_n - \bar{x} \end{bmatrix}$$

Let $Q = X^T X$ be the NxN matrix

$$Q = \begin{bmatrix} x_1 - \bar{x} & x_2 - \bar{x} & ... & x_n - \bar{x} \end{bmatrix} \begin{bmatrix} (x_1 - \bar{x})^T \\ (x_2 - \bar{x})^T \\ \cdot \\ \cdot \\ \cdot \\ (x_n - \bar{x})^T \end{bmatrix} \tag{16}$$

Q is the covariance matrix. Using eigenvalues and eigenvectors of Q, we interpreted that eigenvectors with the greatest eigenvalue corresponds to the dimensions that have the greatest correlation in the data-set. This is the principal component.

*C. **t-SNE***

[14] It is a technique that is used to visualize high-dimensional data in 2-D or 3-D. It is a variation of SNE and is easy to optimize.

*1) **Stochastic Neighbor Embedding (SNE):*** First step here is to convert the Euclidean distance in the higher-dimension to conditional probabilities which will represent the similarity. $p_{j|i}$ gives the similarity of data-points x_j and x_i. This means that if neighbours were chosen in proportion to their probability density under a normal curve centered at x_i, then x_j will be a neighbour. If the points are nearer, then $p_{j|i}$ is large and if they are far apart, $p_{j|i}$ is infinitesimal. Mathematical expression for this is,

$$p_{j|i} = \frac{exp(-(\frac{||x_i-x_j||^2}{2\sigma_i^2}))}{\sum_{k\neq i} exp(-(\frac{||x_i-x_k||^2}{2\sigma_i^2}))} \tag{17}$$

where σ_i is the variance of normal distribution centered at x_i. $p_{i|i} = 0$. y_i and y_j are the low dimensional counter part of x_i and x_j. Similar conditional probability in low-dimensional space is denoted by $q_{j|i}$. Suppose the variance of the normal distribution for the equation $q_{j|i}$ is set to $\frac{1}{\sqrt{2}}$. So,

$$q_{j|i} = \frac{exp(-(||y_i - y_j||)^2}{\sum_{k\neq i} exp(-(||y_i - y_k||^2)} \tag{18}$$

Again, we set $q_{j|i} = 0$. If the points y_i and y_j can exactly find the similarity between the points x_i and x_j, then $p_{j|i}$ and $q_{j|i}$ will be same. So the aim of SNE is to achieve this task. It is done through Kulback-Leibler divergence. Using gradient descent algorithm, SNE minimizes sum of this term.

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}} \tag{19}$$

Where P_i and Q_i are the conditional probabilities of all the other data-points when x_i and y_i are given. Now, only the variance σ_i of the normal distribution which is centered over each x_i needs to be known. As the density of the data will vary, there won't be any single value for σ. Small value of σ is used in dense regions. Any value of σ induces a probability distribution P_i over all the other points. P_i has an entropy that is proportional to σ_i. SNE uses a binary search mechanism for σ_i that will lead to a P_i and has a fixed perplexity. Defined as,

$Perp(P_i) = 2^{H(P_i)}$, where $H(P_i)$ is the entropy of P_i.

$$H(P_i) = -\sum_j P_{j|i} log2 P_{j|i} \tag{20}$$

The minimization of (5) is done through gradient descent.

$$\frac{\delta C}{\delta y_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j) \tag{21}$$

*2) **t-Distributed Stochastic Neighbor Embedding (t-SNE):*** Eventhough SNE gives us good visualization of the data, it has an issue called "crowding problem". This means that, the area of the 2-D map that is available to accommodate distant points won't be large enough when we compare with the area available to accommodate nearer points. t-SNE will take care of all these issues. The cost function used here have two main differences:
(1). A symmetrized version of the SNE cost function is used.
(2).To calculate similarity, it uses Students-t distribution rather than normal distribution
Instead of minimizing the sum of Kullback-Leibler divergence, it tries to minimize single Kullback-Leibler between a joint probability P, in high-dimension and corresponding Q in low-dimension.

$$C = KL(P||Q) = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}} \tag{22}$$

Here we have $p_{j|i} = p_{i|j}$ and $q_{j|i} = q_{i|j}$, so this is symmetric SNE. Pairwise similarities in low-dimension,

$$q_{i|j} = \frac{exp(-(||y_i - y_j||)^2}{\sum_{k \neq l} exp(-(||y_l - y_k||^2)} \tag{23}$$

Pairwise similarities in high-dimension,

$$p_{i|j} = \frac{exp(-(\frac{||x_i - x_j||^2}{2\sigma^2}))}{\sum_{k \neq l} exp(-(\frac{||x_l - x_k||^2}{2\sigma^2}))} \tag{24}$$

If we have an outlier x_i in high-dimension, then $||x_i - x_j||^2$ is large for all j. So, $p_{i|j}$ is small $\forall$ j. t-SNE overcome this problem by defining the joint probabilities in the high-dimension as $p_{i|j} = \frac{p_{j|i} + p_{i|j}}{2n}$. By doing so, we get that $\sum_j p_{i|j} > \frac{1}{2n}$. By this, each point in the high-dimension will contribute significantly to the cost function. The gradient is given by,

$$\frac{\delta C}{\delta y_i} = 4 \sum_j (p_{i|j} - q_{i|j})(y_i - y_j) \tag{25}$$

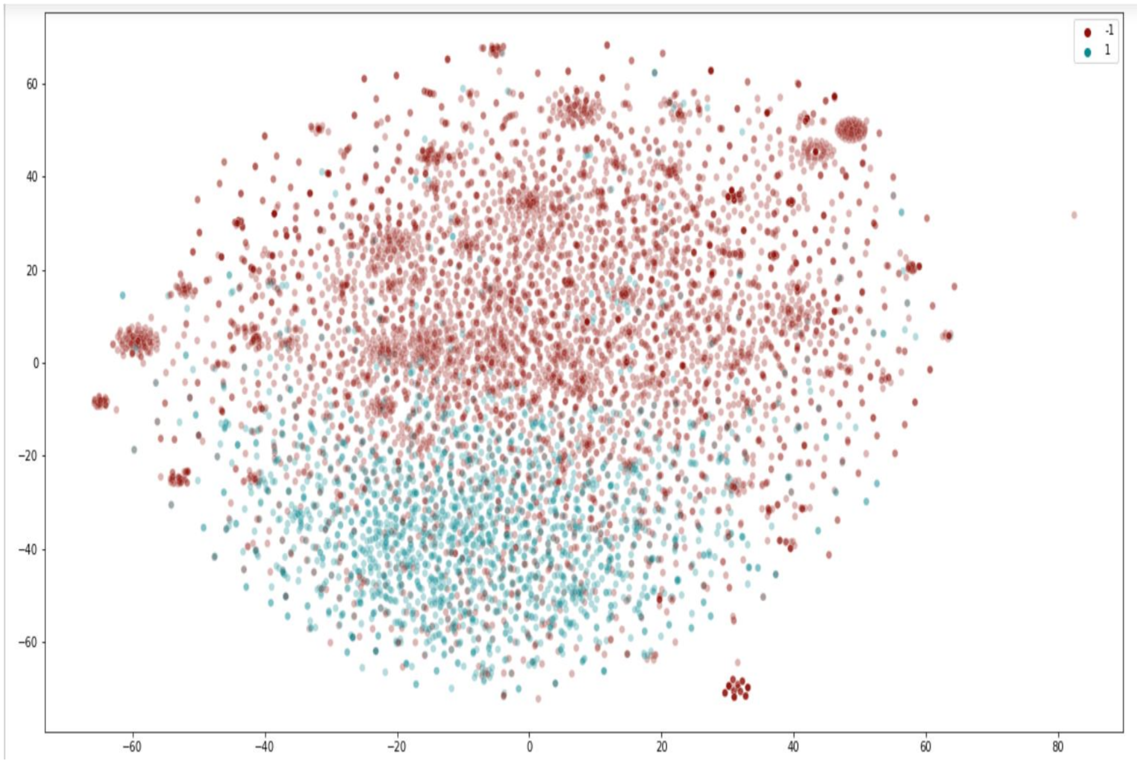


Fig. 4. Data-set t-SNE.

D. *SMOTE*

It is an Oversampling method used to solve class imbalance. It will randomly increase the minority class by replicating. It will synthesis more minority instance between already existing instances. These are done by selecting one or more of random k-nearest neighbour for each minority instances. Algorithm is as follows:
(1). Given the minority class S, for each $y \in S$, the nearest k-neighbours of y are obtained using Euclidean distance of y and every other elements in S.
(2). Sampling rate T is given according to the proportion of imbalance. For each $y \in S$, T elements are selected randomly from nearest k-neighbours. And the set S_1 is made.
(3). For every $y_k \in S_1$, $k = 1,2,3...,T$, the formula for generating new example is,

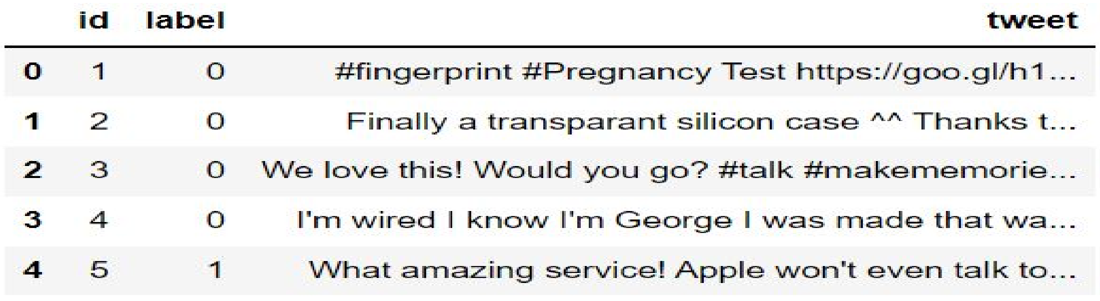
$$y' = y + rand(0,1) * |y - y_k| \tag{26}$$

V. Methodology

The kernel method approach towards classifying sentiment from tweets is done using python3 via jupyter notebook. In this step we will describe each step in constructing the same.

A. *Data-set Description*

The data-set we used has 7920 instances and 3 features, namely id, sentiment and tweets. This data set contains 2 classes namely, 'positive', 'negative'. It is taken from Kaggle. Figures below will give a quick glance to our data-set and data-statistics.



	id	label	tweet
0	1	0	#fingerprint #Pregnancy Test https://goo.gl/h1...
1	2	0	Finally a transparant silicon case ^^ Thanks t...
2	3	0	We love this! Would you go? #talk #makememorie...
3	4	0	I'm wired I know I'm George I was made that wa...
4	5	1	What amazing service! Apple won't even talk to...

Fig. 5. This is an sample from our data-set.

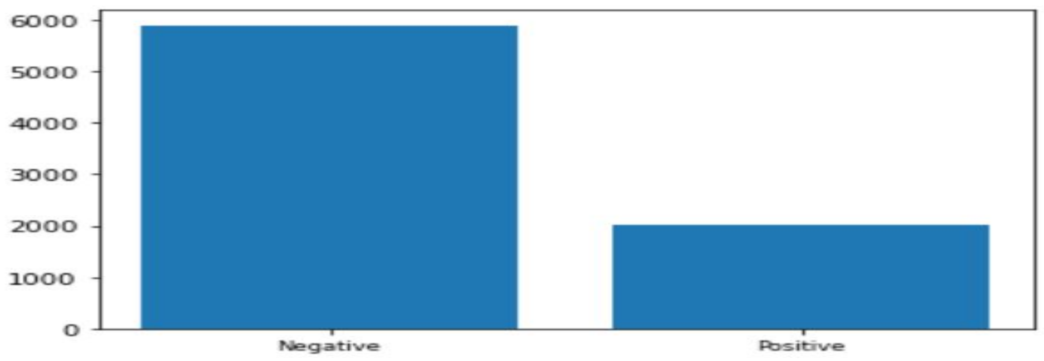


Fig. 6. Data-set class information.

B. Data-Preprocessing

After extracting only two columns that if 'tweets' and 'Labels', from the data-sets we used NLTK library to Lemmetize and remove Stop-words from the transcripts. After cleaning the data, the text is then need to be transformed into meaningful numerical values, so that a computer can understand. For this we used TF-IDF vectorizer before training the model. After this step we visualized our high-dimensional data in 2-D with the help of t-SNE. Fig. 3. shows this.

*1) **Applying SVM:*** We have reduced the dimension using PCA before fitting to our models. The feature dimension reduced from 21616 to 5490 for our data-set. Train test split is then done in the ratio 80:20. We then used SVM and its linear and Gaussian Kernels for classification. Trained our model and predicted the target values of 20% test data instances. The overall accuracy of each model and the class-wise precision, recall, f1-score is also calculated using sklearn's classification-report() function.

VI. Experimental Results and Discussion

The table below shows the evaluation scores for each of our Kernels.

Comparative Performance scores of Kernels using cvxpy				
	Precision	Recall	F1-Score	Accuracy
Linear	0.85	0.85	0.85	0.85
Gaussian	0.83	0.83	0.82	0.83

Comparative Performance scores of Kernels from Sklearn				
	Precision	Recall	F1-Score	Accuracy
Linear	0.85	0.85	0.85	0.85
Gaussian	0.87	0.87	0.87	0.87

From the above table, it is clear that SVM with Gaussian Kernel performs well.

VII. Conclusion

In this work, we tried to compare some of SVM Kernels and evaluated their performance based on different evaluation metrics such as f1-Score, Precision etc... We got an overall accuracy of 87% for Gaussian Kernel. In near future, we will extend this task to a web based application. We will add more algorithms from deep learning and collect more data to get better results.

References

[1] P. Koehn, *Statistical machine translation*, Cambridge University Press, 2009. N. Sadman, K. D. Gupta, A. Haque, S. Poudyal, and S. Sen

[2] N. Sadman, K. D. Gupta, A. Haugue, S. Poudyal *"Detect review manipulation by leveraging reviewer historical stylometrics in amazon, yelp, facebook and google reviews,"*, in Proceedings of the 2020 The 6th International Conference on E-Business and Applications, 2020, pp. 42–47.

[3] K. Kourou, T. P. Exarchos, K. P. Exarchos, M. V. Karamouzis, and D. I. Fotiadis, *"Machine learning applications in cancer prognosis and prediction," Computational and structural biotechnology journal*, vol. 13, pp. 8–17, 2015.

[4] AdaHealth. Ada - your health companion. [Online]. Available: https://ada.com/

[5] M. Krallinger, R. A. Erhardt and A. Valencia, *"Text-mining approaches in molecular biology and bio-medicine,"* Drug discovery today, vol. 10, no. 6, pp. 439–445, 2005.

[6] M. Khachidze, M. A. Tsintsadze and M. Archuadze, *"Natural language processing based instrument for classification of free text medical records,"* BioMed research international, vol. 2016, 2016.

[7] M. S. Ong, F. Magrabi and E. Coiera, *"Automated identification of extreme-risk events in clinical incident reports,"* Journal of the American Medical Informatics Association, vol. 19, no. e1, pp. e110–e118, 2012.

[8] E. Frank and R. R. Bouckaert, *"Naive bayes for text classification with unbalanced classes,"*in European Conference on Principles of Data Mining and Knowledge Discovery. Springer, 2006, pp. 503–510.

[9] L. E. Peterson *"K-nearest neighbor,"*Scholarpedia, vol. 4, no. 2, p. 1883, 2009.

[10] Vapnik, V. N and A. Y. Chervonenkis, (1964), *"A note on one class Perceptrons."*Automation and remote control, Vol. 25

[11] Vapnik, A. Liaw and M. Wiener et al., *"Classification and regression by randomforest,"*R news, vol. 2, no. 3, pp. 18–22, 2002.

[12] N. Sadman, S. Tasneem, A. Haque,M. M. Islam,M. M. Ahsan and K. D. Gupta, *"Can NLP techniques be utilized, as a reliable tool for ,medical field?"*- Building a NLP Framework to Classify Medical Reports," 2020 11th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2020, pp. 0159-0166, doi: 10.1109/IEMCON51383.2020.9284834.

[13] N. Sadman, M. H. Rahman, S. Tasneem,M. A. Haque,and K. D. Gupta, *"Recommend, Speciality Doctor from Health Transcription: Ensemble, Machine Learning ,Approach,"* 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC), 2021, pp. 0967-0972, doi: 10.1109/CCWC51732.2021.9376111.

[14] Laurens. van. der. Maaten, Geoffrey. Hinton *"Visualizing Data using t-SNE"* Journal of Machine Learning Research 9 (2008) 2579-2605, Submitted 5/08; Revised 9/08; Published 11/08.

[15] Feng. M, G. Wu. *"A distributed chinese Naive Bayes classifier based on word embedding"* International Conference on Machinery Materials and Computing Technology (ICMMCT 2016), 2016, pp. 1121-1127