Performance Analysis of Different Machine Learning Approaches for Single Modal Facial Expression Detection

Supervisor: Dr. Md. Ashraful Alam

Surovi Zinia	13201058
Aliza Ibn Azmol	13321063
Saiful Islam	13121125

Department of Computer Science and Engineering,

BRAC University.

Submitted on: 13th August, 2018

Declaration

We hereby declare that this thesis is based on results obtained from our own work. Due acknowledgement has been made in the text to all other material used. This thesis, neither in whole nor in part, has been previously submitted to any other University or Institute for the award of any degree or diploma.

Signature of Supervisor **Signature of Authors**

Dr. Md Ashraful Alam

Surovi Zinia (13201058)(CSE)

Aliza Ibn Azmol (13321063)

Saiful Islam (13121125)

Acknowledgement

In the beginning we would like to thank the Almighty ALLAH the creator and protector of this world. Our gratefulness to him is beyond limit as He provided us with wellbeing and abilities, for which we could complete our thesis successfully. May He be glorified and exalted for the blessings that He has bestowed upon us in both spiritual and worldly term.

Even though thank you cannot express the unconditional support and guidance Dr. Md Ashraful Alam, our thesis supervisor, provided us with throughout our project, we would like to thank Sir from the bottom of our heart. Without Dr. Md Ashraful Alam Sir, our thesis project wouldn't have seen the light of success.

We can never forget the help we received from the BRAC University Faculty Staffs of the Computer Engineering & Engineering Department. They have been with us throughout the whole journey and were right beside us for guidance any time we needed them, particularly in developing our basic knowledge and enhancing our capabilities. Thank you for always being there.

At the end, we would be proud to express our thankfulness and appreciation to our respectable parents, brothers and sisters for their motivation and support. We are grateful to all of our family like friends who helped to complete this project.

Table of Contents

List of Figures

List of Tables

Abstract

Facial expression detection plays a pivotal role in the studies of emotion, cognitive processes, and social interaction. This has potential applications in different aspects of everyday life .For Example, real time face detection, sentiment analysis, CCTV violence prediction. In this thesis, we investigate and analyze the performance of different machine learning approaches for single modal type facial expression detection. With this proposed model, it is observed that the feature extraction techniques incorporated in this work performs better in recognizing disparate expressions than feeding unprocessed raw dataset to the networks. Moreover, this study used Japanese Female Facial Expression (JAFFE) to demonstrate the comparative performance of different classical classifiers and neural network-based approaches and how viable they are in the detection of facial expression from single modal information. Hence this kind of models increase the advancement of facial recognition for more future purposes .Therefore, the proposed model proves the feasibility of computer vision based facial expression recognition for practical applications like surveillance and Human Computer Interaction (HCI). In this system, Principal Component Analysis (PCA) and t-distributed Stochastic Neighbor Embedding (t-SNE) have been used to solve the dimensionality reduction and visual representation of the feature components in a 2D feature space. For classification and recognition tasks we used different classification algorithms like K-nearest Neighbor (KNN), Support Vector Machines (SVM), Gaussian Naïve Bayes, Random Forest, Extra Tree, Ensemble machines and vanilla neural networks. To use the total dataset on this algorithm we used 80% training and 20% testing of the total dataset. Finally the best accuracy result was given by Artificial Neural Network (ANN) which was 90.63 % from the proposed model.

Keywords: Artificial Neural Network(ANN), Logistic Regression, SVM (Support Vector Machine), Gaussian Naïve Bayes , K-nearest Neighbor(KNN), Principal Component Analysis (PCA), t-distributed Stochastic Neighbor Embedding (t-SNE)

Chapter 01

Introduction

1.1 Motivation

Pictures are the outline of the circumstances of our surrounding. We realized that emotion recognition and tracking can be really beneficial in healthcare, social media and market research areas. The complexity of recognizing emotions of images with machine learning technique is significant. It's not all that long ago that Facebook moved on from having only a straightforward like button to allowing users the option to select from a number of emotional reactions. Therefore, finding the best machine learning algorithm for emotion detection gave us a purpose to conduct a research to have the comparative study of different approaches in order to find a better image recognition methodology.

1.2 Aims and Objectives

The primary aim of our research work is to find the best machine learning algorithm to detect emotion from 2D still images. There has been some works done in the field of emotion detection. Each one of them is unique in the methodologies which have been used in the respective works. This research work focuses on the comparative study of different approaches to obtain a better set of methods to achieve better emotion detection.

1.3 Thesis Outline

The rest of the thesis is organized as follows:

- Chapter 2 discusses the Literature Review of related works in this field.
- Chapter 3 provides the Background study in details including the techniques used in the system.
- Chapter 4 describes the proposed model along with implementation details.
- Chapter 5 presents the results of the experiment along with performance analysis and comparisons.
- Chapter 6 concludes the paper specifying the limitations and challenges while planning future development of the project.

Chapter 02

Literature Review

2. 1 Background Study

The detection and quantification of facial nuances provide an important behavioral measure for the study of emotions, cognitive processes [1] and thus automatic facial expression recognition systems can render a non-invasive method to discern the emotional activity of a subject. With the availability of low cost imaging and computational devices, automatic facial recognition systems are extremely viable in various everyday situations like operator fatigue detection in industries, user mood detection in human computer interaction (HCI) and possibly in identifying suspicious persons in airports, railway stations and other places with higher threat of terrorism attacks. In these cases, machines can compete with or in specific cases, even outperform human detection of facial expression. It is also a significant step towards a fully autonomous Human Computer Interaction (HCI) system [2] as facial expressions play a fundamental role in conveying human emotions. Any natural HCI system thus should take advantage of the human facial expressions. It is a topic of immense controversy in psychology and behavioral science literature regarding whether facial expressions are universally constant or not and regarding whether facial expressions occur involuntarily or voluntarily [3]. Extreme theorist in the early days are now gradually accepting the interactionist perspectives integrating solid evidence for both universality and cultural specificity [4].Although facial expression is biased by cultural specificity, research has shown that the expressions are almost always infallibly recognized by people universally at a rate greater than that allowed by coincidence factor and hence, technically the inner nuances of facial expressions are universal. At the same time, research also shows that cultural exposure elevates the chances of correct recognition of facial expressions indicating cultural dependence [5].

Until recently, there were only two options for correct recognition of facial expressions: human observer based coding system and electromyography (EMG) based systems [6]. Human observer based methods are time consuming to learn and apply, difficult to standardize and often proves to be economically infeasible. The other approach, facial EMG, requires placement of sensors on

the face, which may inhibit certain facial actions and rules out its use for naturalistic observation. A promising alternative is automated facial image analysis using computer vision. The research in computer vision based recognition of facial expressions has progressed for long irrespective of the psychological debate. The primary inspiration of such research efforts has been the human ability to recognize facial expressions by just looking at still or video images with a high rate of correct recognition. The potential benefits of computer recognition of facial expressions in security applications and HCI have been the motivations in most of the cases. There are mainly two different approaches that are commonly being used in computer vision based facial expression recognition: recognition using 2D still images and recognition using image sequences. Approaches using image sequence often apply optical flow analysis to the image sequence and use pattern recognition tools to recognize optical flow patterns associated with particular facial expression. This approach requires acquisition of multiple frames of images to recognize expressions and thus has limitations in real-time performance and robustness. Facial expression recognition using still images often use feature based methods [2] for recognition and thus have faster performance but the challenge in this approach is to develop a feature extraction method that works well regardless of variations in human subjects and environmental conditions. Gabor filter banks are reasonable models of human optical processing in primary visual cortex and are one of the most successful approaches for processing images of the human face [7]. Gabor wavelet based facial expression coding system and show that their representation method has a high degree of correlation with the human semantic ratings. In ref [8], Gabor filter banks based facial expression coding for feature extraction and multilayer perceptron (MLP) based feature classification is reported to have performed better than geometric feature based facial expression recognition.

In our work, we used the feature extraction method proposed in [9]. Principal component analysis (PCA) and t-distributed Stochastic Neighbor Embedding (t-SNE) are used for dimensionality reduction and visual representation of the feature components in a 2D feature space. Different classification algorithms like K-nearest Neighbor (KNN), Support Vector Machines (SVM), Gaussian Naïve Bayes, Random Forest, Extra Tree, Ensemble machines and vanilla neural networks have been widely used for classification and recognition tasks. The use of neural networks in face recognition has addressed several problems: gender classification,

face recognition and classification of facial expressions [10]. There are different architectures of neural networks each having their own strengths and drawbacks. Good performance of a given architecture in a particular problem does not ensure similar results in a different problem. In this thesis work, benefits of intricate feature extraction and application of Artificial Neural Network (ANN) are explored for recognition of facial expression rather than MLP. By using Japanese Female Facial Expression (JAFFE) database for training and testing, the performance of different classifiers has been compared before and after performing deliberate feature extraction procedure.

2.2 Modes of Emotion Perception

Emotions can be perceived through visual, auditory, olfactory, and physiological sensory stimulations. Nonverbal actions can provide social partners with information about subjective and emotional states. The quantification of emotion can be recapitulated from single modal information like images of different facial expression or from the combination of two or more types of information. In our thesis however, we have emphasized on extracting emotion from single modal information- in this case, 2D images of facial expression.

2.3 Different Approaches to discern Facial Expression

Over the years different approaches like signal processing, geometric shape drawing and machine learning have been adopted by the researchers for training machines to differentiate between different emotional states from various types of information like speech, text, audio, video and a combination of those.

2.3.1 Lexicon Based Approaches

This technique mainly incorporates the combination of domain knowledge, the semantic and syntactic characteristics of language to perceive certain types of emotion [11]. In this approach, knowledge-based resources such as WordNet, SenticNet[12], ConceptNet, and EmotiNet[13] are basically used. One of the merits of this approach is the accessibility and economic feasibility brought about by the large availability of such knowledge-based resources. A limitation of this technique on the other hand, is its ineptitude of handling subtle nuances and complex linguistic rules. Knowledge-based techniques can be mainly classified into two categories: dictionary-based and corpus-based approaches. Dictionary based approaches generally execute words and

syllables from a definite dictionary containing predefined label of emotion corresponding to each word. These approaches do not consider the context and other subtlety entwined with linguistics. On the contrary, corpus-based approaches start with a seed list of opinion or emotion words and expand the database by finding other words with context-specific characteristics in a large corpus. While corpus-based approaches take contexts into account, their performance still vary in different domains since a word in one domain can have a different orientation in another domain [14].

2.3.2 Statistical Approaches

This approach mainly incorporates different types of supervised and unsupervised machine learning techniques to learn the internal difference between different emotions. With the influx of GPU computing and faster memory, now these machine learning algorithms out performs traditional signal processing or geometry based techniques by a large margin. In this work we applied Support Vector Machine (SVM), K-nearest Neighbor (KNN), Gaussian Naïve Bayes, Decision Tree method, Random Forest, Extra Tree method and some neural network based deep learning methods to discern different emotions from single modal information. Well-known deep learning algorithms include different architectures of Artificial Neural Network (ANN) such as Convolutional Neural Network (CNN), Long Short-term Memory (LSTM), and Extreme Learning Machine (ELM). The popularity of deep learning approaches in the domain of emotion recognition maybe mainly attributed to its success in related applications such as in computer vision, speech recognition, and Natural Language Processing (NLP)[15-17].

2.3.3 Hybrid Approaches

Hybrid approaches are basically a combination of lexicon based techniques and statistical methods, which executes interdependent attributes from both techniques. Some of the works that have applied an ensemble of lexicon based elements and statistical methods. Since hybrid techniques gain from the benefits offered by both knowledge-based and statistical approaches, they tend to have better classification performance as opposed to employing lexicon-based or statistical methods independently. A downside of using hybrid techniques however, is the computational complexity often suffers from combinatorial explosion.

Chapter 03

Proposed System Description

Since we have taken statistical approaches for the demonstration of emotion detection from facial expression, at first, we selected suitable dataset for that. Japanese Female Facial Expression (JAFFE) dataset has been used to show how autonomous machine learned emotion recognition works. There are 213 photos of models rendering 7 different emotions. To visualize high dimensional feature space of each sample we applied various dimensionality reduction techniques and for detecting and differentiating between discerning facial expressions we applied various classifiers on the dataset. We stacked up these different techniques to see how they perform compared to each other. Our demonstration can be surmised in the following way:

- Dimensionality Reduction Techniques
  i) Principle Component Analysis (PCA)
  ii) Locally Linear Embedding (LLE)
  iii) Linear Discriminant Analysis (LDA)
  iv) t-distributed Stochastic Neighbor Embedding (t-SNE)
- Network Analysis and Classification
  i) Supervised Learning
  ii) Unsupervised Learning

We mainly emphasized on the supervised learning part.

3.1 Dimensionality Reduction Technique

3.1.1 Principle component analysis (PCA)

PCA is a powerful linear unsupervised pattern recognition system that reduces the dimensionality of a multivariate problem and helps to visualize the different categories of taste and odor profiles by highlighting similarities and differences between sample clusters [18].

In multivariate system it's a problem to visualize the data. But more than two variables are difficult to visualize the data in a graph. This feature extraction method consists of projecting the N dimensional data set (N is the number of sensor) in a new base of same dimension$N$.

Mathematical Representation

PCA is mathematically defined as an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by some projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate.

Consider a data matrix, $X_{n\times 2}$ where *n* rows represent a different sensor output of the experiment, and each of the columns gives a kind of feature (say, the results for a particular unit).

In PCA Eigen vector corresponding to large eigenvalue is called 1st principle component and other called 2nd principle component. Eigenvalues are finding from the covariance matrix of this data plot.

$$Covariance\ matrix = (x - \mu) \times (x - \mu)^T \quad (1)$$

$$\mu = mean\ value$$

$$\text{So,} = X_{2\times n} \times X_{n\times 2} \times N^{-1} \quad (2)$$

Where N^{-1} is the Data point; two variable presents

If the no of variable is N then the covariant matrix dimension will be $N \times N$.

So here,

$$cov = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \quad (3)$$

$$Eigenvalues\ \ Ax = \lambda x \quad (4)$$

Determinant matrix for eigenvalues:

$$\begin{bmatrix} A - \lambda & C \\ B & D - \lambda \end{bmatrix} = 0 \quad (5)$$

From this matrix we will get the value of λ:

So finally,

$$\begin{bmatrix} A & C \\ B & D \end{bmatrix} \times \begin{pmatrix} x1 \\ x2 \end{pmatrix} = \lambda \begin{pmatrix} x1 \\ x2 \end{pmatrix} \quad (6)$$

This creates a linear relationship and when they are plotted, they reduce the dimension.

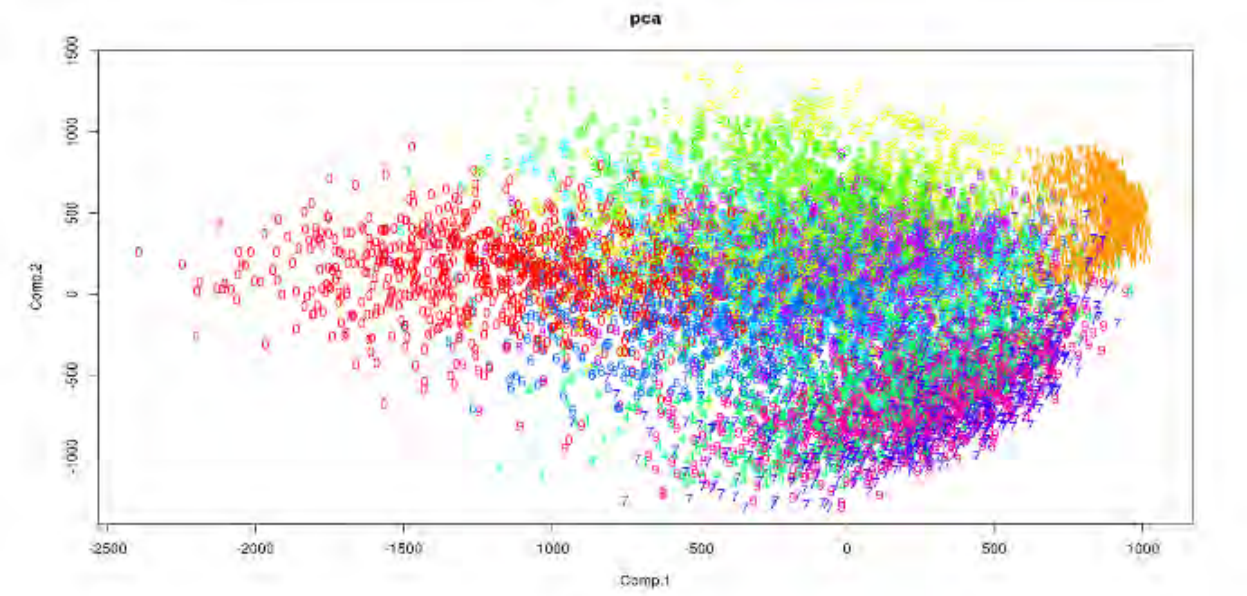


Figure 1: Principal component analysis (PCA).

Limitations of PCA

PCA is a linear algorithm. It will not be able to interpret complex polynomial relationship between features. On the other hand, t-SNE is based on probability distributions with random walk on neighborhood graphs to find the structure within the data.

A major problem with, linear dimensionality reduction algorithms is that they concentrate on placing dissimilar data points far apart in a lower dimension representation. But in order to represent high dimension data on low dimension, non-linear manifold, it is important that similar datapoints must be represented close together, which is not what linear dimensionality reduction algorithms do.

Local approaches seek to map nearby points on the manifold to nearby points in the low-dimensional representation. Global approaches on the other hand attempt to preserve geometry at all scales, i.e mapping nearby points to nearby points and far away points to far away points.

It is important to know that most of the nonlinear techniques other than t-SNE are not capable of retaining both the local and global structure of the data at the same time.

3.1.2 Locally Linear Embedding (LLE)

Locally linear embedding (LLE) developed by Roweis and Saul in 2000 [19]is a promising method for the problem of nonlinear dimensionality reduction of high-dimensional data. Unlike classical linear dimensionality reduction methods, LLE provides information that can reveal the intrinsic manifold of data. It assumes that each data point and its neighbors lie on a locally-linear patch, and then applies this patch in a low space to generate data configuration. LLE recovers global non-linear structures from locally-linear fits.

LLE maps a input data set $X = \{x_1, x_2, \dots \dots x_n\}$, $x_i \epsilon R^d$ globally to a output data set

$Y = \{y_1, y_2, \dots \dots \dots \dots y_n\}$, $y_i \epsilon R^m$ where $m \ll d$. Assuming the data lies on a nonlinear manifold which locally can be approximated linearly, the algorithm has three sequential steps:

Step 1. Determining neighbors: The K closest neighbors are selected for each point using a distancemeasure such as the Euclidean distance.

Step 2. Calculating reconstruction weights: In order to determine the value of the weights, the reconstruction errors are measured by the cost function:

$$\epsilon(W) = \sum_{i=1}^{n} \left| x_i - \sum_{j=1}^{n} w_{ij} x_j \right|_2 \qquad (7)$$

The weight w_{ij} are determined by minimizing the cost function defined in equation (A) subject to two constraints $\sum_j W_{ij} = 1$ and $w_{ij} = 0$ if x_j is not neighbour of x_i . The weights are then determined by a least-squares minimization of the reconstruction error.

Step 3. Finding lower-dimensional embedding Y: Define a cost embedding function:

$$\Phi(Y) = \sum_{i=1}^{n} \left| y_i - \sum_{j=1}^{n} w_{ij} y_j \right|_2 \qquad (8)$$

The lower-dimensional coordinate are computed by minimizing the cost function defined in Equation (B) subject to two constraints: $\sum_i y_i = 0$ and $\sum_i \frac{y_i y_j{}^T}{N} = I$, where I is a $m \times n$ identity matrix. The above optimization problem can be solved by transforming it to an eigenvalue problem.

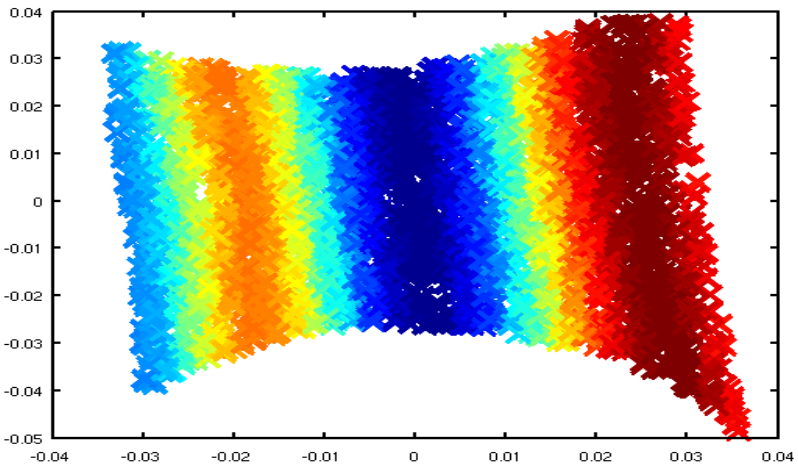


Figure 2: Locally linear embedding (LLE).

3.1.3 Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) commonly used techniques for data classification and dimensionality reduction. Linear Discriminant Analysis easily handles the case where the within-class frequencies are unequal and their performances have been examined on randomly generated test data. This method maximizes the ratio of between-class variance to the within-class variance in any particular data set thereby guaranteeing maximal separability.

In PCA, the shape and location of the original data sets changes when transformed to a different space where as LDA doesn't change the location but only tries to provide more class separability and draw a decision region between the given classes. This method also helps to better understand the distribution of the feature data.

Mathematical Representation of LDA

In LDA class dependent transformation is applied which involves maximizing the ratio of between class variance to within class variance. So, in LDA maximization of class separation is

wanted and two different matrices are formed. One is within class matrix and another is between class matrixes. These two matrix helps to maximize the ratio between class scatter and within class scatter.

Between classes scatter matrix is defined by:

$$S_B = \sum_{i=1}^{c} (\mu_i - \mu)(\mu_i - \mu)^T \quad (9)$$

$$where,$$

$$\mu i = Mean\ of\ particular\ class\ Xi$$

$$Ni = No\ of\ sample\ in\ particular\ class\ Xi$$

$$\mu = over\ all\ mean\ of\ the\ data.$$

Within class scatter matrix is defined by:

$$S_w = \sum_{i=1}^{c} \sum_{X_k \in X_i}^{1} (X_k - \mu_i)(X_k - \mu_i)^T \quad (10)$$

Here, S_B and S_w have the same dimension as the scatter matrix. So, cluster between two class become more and within class scatter become less.

S_B can be thought of as the covariance of data set whose members are the mean vectors of each class. the optimizing criterion in LDA is the ratio of between-class scatter to the within-class scatter. As defined earlier, the solution obtained by maximizing this criterion defines the axes of the transformed space. If the LDA is a class dependent type, for *L*-class *L*separate optimizing determinant matrix are required for each class.

If the covariance matrix is

$$cov = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \tag{11}$$

Determinant matrix for eigenvalues

$$\begin{bmatrix} A-\lambda & C \\ B & D-\lambda \end{bmatrix} = 0 \tag{12}$$

From this matrix we will get the value of λ.

Here, the determinant of the matrix is computed using the equation $A\&B$ and it computed as,

$$Detmatrix = \ inv(S_w) \times \ S_B \tag{13}$$

A set of these Eigen vectors whose corresponding Eigen values are non-zero are all linearly independent and are invariant under the transformation. Thus, any vector space can be represented in terms of linear combinations of the Eigen vectors. A linear dependency between features is indicated by a zero Eigen value. To obtain a non-redundant set of features all Eigen vectors corresponding to non-zero Eigen values only are considered and the ones corresponding to zero Eigen values are neglected.

For any L-class problem we would always have $L-1$non-zero Eigen values.

Once the transformations are completed using the LDA transforms, Euclidean distance is used to classify data points. Euclidean distance is computed using the equation (D) where $\mu_{ntran}$is the mean of the transformed data set, n is the class index and x is the test vector. Thus, for n classes, n Euclidean distances are obtained for each test point.

$$dist-n = (transform\ n\ space)^T\ x - \mu_{ntrans} \tag{14}$$

The smallest Euclidean distance among the n distances classifies the test vector as belonging to class.

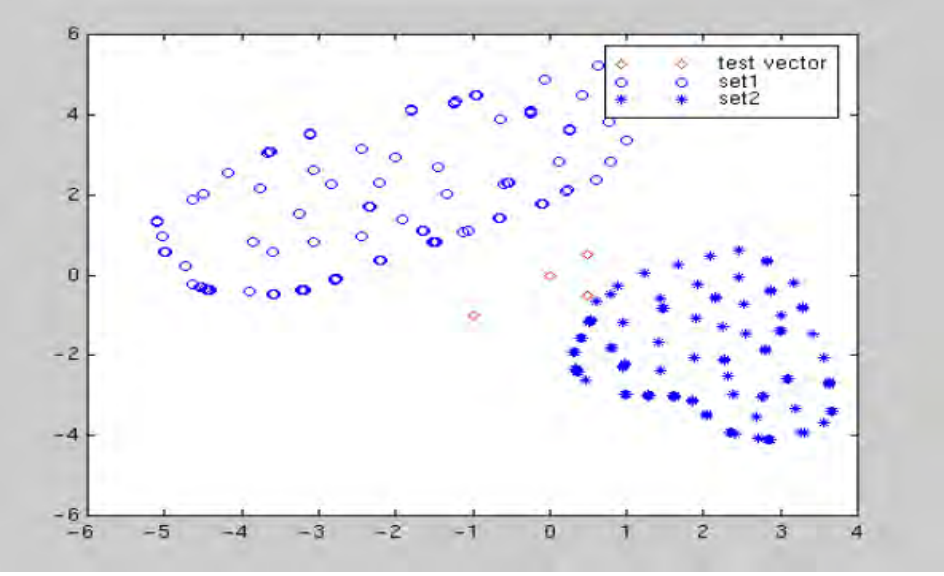


Figure 3: Linear Discriminant Analysis (LDA).

3.1.4 (t-SNE) t-Distributed Stochastic Neighbor Embedding

(t-SNE) t-Distributed Stochastic Neighbor Embedding is a non-linear dimensionality reduction algorithm used for exploring high-dimensional data. It maps multi-dimensional data to two or more dimensions suitable for human observation.

t-SNE is a non-linear dimensionality reduction algorithm finds patterns in the data by identifying observed clusters based on similarity of data points with multiple features. But it is not a clustering algorithm it is a dimensionality reduction algorithm. This is because it maps the multi-dimensional data to a lower dimensional space, the input features are no longer identifiable. Thus, you cannot make any inference based only on the output of t-SNE. So essentially it is mainly a data exploration, dimension reduction and visualization technique.

But t-SNE can be used in the process of classification and clustering by using its output as the input feature for other classification algorithms.

Mathematical representation of t-SNE

Stochastic Neighbor Embedding (SNE) starts by converting the high-dimensional Euclidean distances between data points into conditional probabilities that represent similarities. The similarity of data point x_i to data point x_j is the conditional probability. $P_{j|i}$, x_i would pick x_j as

its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at x_i.

For nearby data points, $P_{j|i}$ is relatively high, whereas for widely separated data points, $P_{j|i}$ will be almost infinitesimal (for reasonable values of the variance of the Gaussian, σ_i). If x is the discrete data points, the variance of a set of n equally likely values can be written as

$$varience\sigma_i = Var(x) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2 \tag{15}$$

Where μ is the expected value

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{16}$$

Mathematically, the conditional probability $P_{j|i}$ is given by –

$$P_{j|i} = \frac{\exp\left(-\frac{\left||x_i - x_j|\right|^2}{2\sigma^2}\right)}{\sum_{k\neq i}\exp(-\frac{\left||x_i - x_j|\right|^2}{2\sigma_i^2})} \tag{17}$$

Where σ_i is the variance of the Gaussian that is centered on data point x_i.

The algorithm starts by converting the shortest distance (a straight line) between the points into probability of similarity of points. Where, the similarity between points is: the conditional probability that x_i would pick x_j as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian (normal distribution) centered at x_i.

For the low-dimensional counterparts y_i and y_j of the high-dimensional data points x_i and x_j it is possible to compute a similar conditional probability, which we denote by $q_{j|i}$.

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

Here, $pi|i$ and $pj|j$ are set to zero as we only want to model pair wise similarity.In simple terms step 1 and step2 calculate the conditional probability of similarity between a pair of points in

- High dimensional space
- In low dimensional space

To understand the underlying procedure, we can attempt to map 3D space to 2D space. Here, step1 and step2 are doing is calculating the probability of similarity of points in 3D space and calculating the probability of similarity of points in the corresponding 2D space. Logically, the conditional probabilities $p_{j|i}$ and $q_{j|i}$must be equal for a perfect representation of the similarity of the data points in the different dimensional spaces, i.e. the difference between $p_{j|i}$ and $q_{j|i}$must be zero for the perfect replication of the plot in high and low dimensions. By this logic SNE attempts to minimize this difference of conditional probability.

The remaining parameter to be selected is the variance $\sigma_i$of the student's t-distribution that is centered over each high-dimensional data point x_i . It is not likely that there is a single value of σ_i that is optimal for all data points in the data set because the density of the data is likely to vary. In dense regions, a smaller value of σ_i is usually more appropriate than in sparser regions. Any particular value of $\sigma_i$induces a probability distribution, P_i over all of the other data points.

This distribution has an entropy which increases σ_i as increases. t-SNE performs a binary search for the value of σ_i that produces a P_i with a fixed perplexity that is specified by σ_i the user. The perplexity is defined as

$$P_{erp}(P_i) = 2^{H(p_i)} \tag{19}$$

Where $H(P_i)$ is the Shannon entropy of P_i

$$H(P_i) = -\sum_j P_{j|i} \log_2 P_{j|i} \tag{20}$$

The perplexity can be interpreted as a smooth measure of the effective number of neighbors. The performance of SNE is fairly robust to changes in the perplexity, and typical values are between 5 and 50.

The minimization of the cost function is performed using gradient decent. And physically, the gradient may be interpreted as the resultant force created by a set of springs between the map point y_i and all other map points y_j . All springs exert a force along the direction ($y_i - y_j$).

The spring betweeny_i and y_j repels or attracts the map points depending on whether the distance between the two in the map is too small or too large to represent the similarities between the two high-dimensional data points. The force exerted by the spring between y_i and y_j is proportional to its length, and also proportional to its stiffness, which is the mismatch $(pj|i - qj|i + p\,i|\,j - q\,i|\,j)$ between the pairwise similarities of the data points.

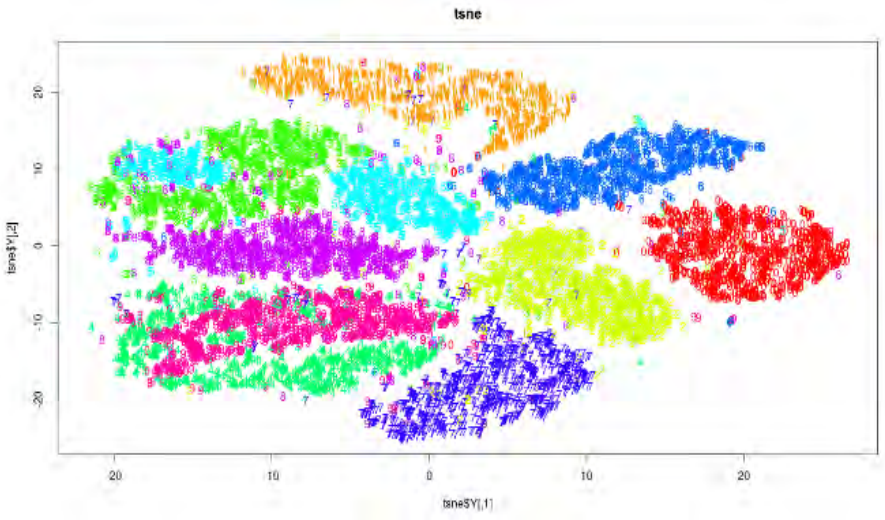


Figure 4: t-Distributed Stochastic Neighbor Embedding (t-SNE).

3.2 Network Analysis and Classification

Network analysis provides the crucial relationships and associations between many objects of different types that are not apparent from isolated pieces of information. There are various techniques employed for the purpose.

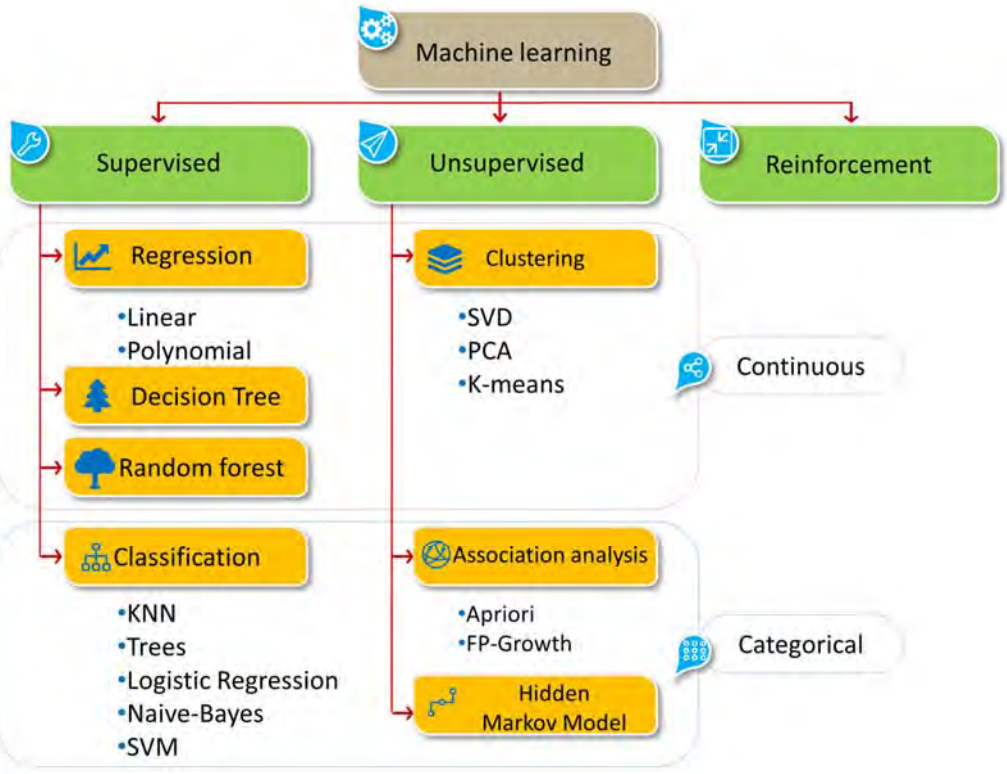
- Supervised approach : Where training data set is pre-labeled
- Unsupervised approach : Where training data set is not pre-labeled

In our thesis we have mainly taken the first route to preprocess our data set. Supervised learning can roughly be divided into two main categories

a. Classical classifiers (Support Vector Machine, k-Nearest Neighbor, Naïve Bayes method)

b. Artificial neural network approach (ANN)

The classical classification methods are mainly hard coded procedures based on statistics. Support Vector Machines, k-Nearest Neighbor, Logistic Regression classifiers, Naïve Bayes method etc. fall under the category of classical classification approaches.



:

Figure 5: Different Machine Learning Approaches for Classification[20].

The artificial neural network (ANN) is the best known and most evolved analysis techniques. An artificial neural network, often just called a neural network, is a mathematical model inspired by biological neural networks. A neural network consists of an interconnected group of artificial neurons, and it processes information using a connectionist approach to computation. In most cases a neural network is an adaptive system that changes its structure during a learning phase. Neural networks are used to model complex relationships between inputs and outputs or to find patterns in data. Various instrument-training methods are employed through pattern-recognition algorithms that look for similarities and differences between identification elements of known patterns found in an analyte-specific reference library. The training process requires a discrete amount of known sample data to train the system and is very efficient in comparing unknown samples to known references. The result of ANN data analysis usually is in the form of a percentage match of identification elements in the sample with those of patterns from known sources in the reference library.

3.2.1 KNN classifier

k-nearest neighbors is a simple algorithm that can be used for solving both classification and regression problems. It has been used in statistical estimation and pattern recognition already in the beginning of 1970's. KNN is simple to understand and is one of the most widely used methods for its faster training phase. It is a nonparametric approach where it does not make any assumption on the underlying data distribution. This is convenient since most of the practical data does not get along with the theoretical assumptions (linearly separable, Gaussian mixture etc.). KNN is a lazy algorithm which means that it avoids using training data points for generalization. It also means that KNN keeps all the training data which translates to a faster training phase. But the whole training data set or at least a subset of that is required to decide which makes the testing period longer.

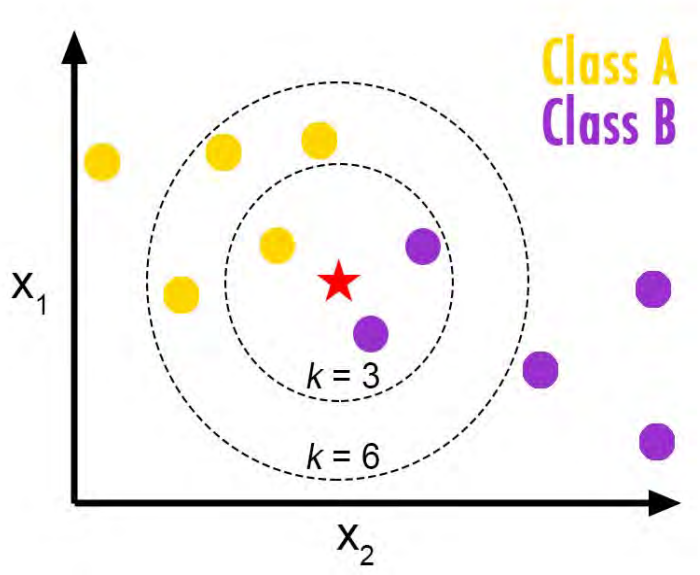


Figure 6: KNN classification mechanism.[21]

KNN classifier assigns a class label for unknown samples by estimating its k-nearest neighbors based on known samples. KNN assumes that the data is in a feature space. The data can be scalars or multidimensional vectors. Since the points are in feature space, they have a notion of distance. All the training examples are stored in the form of $(x_i, f(x_i))$ where x_i is the n-dimensional input feature vector $(x_{i1}, x_{i2}, x_3 \dots x_n)$ and $f(x)$ is the corresponding output. If there is a query point x_q, which is another vector with n-attributes $(x_{q1}, x_{q2}, x_{q3} \dots x_{qn})$, KNN finds the k training examples that are most similar to it using the standard Euclidean distance as a measure of similarity between each training example x_i and x_q :

$$d(x_i, x_q) = \sqrt{(x_{i1} - x_{q1})^2 + (x_{i2} - x_{q2})^2 + \cdots + (x_{in} - x_{qn})^2} \qquad (21)$$

If the target function is discrete-valued, KNN returns the most common target function value among the neighbors of the query point. A case is classified by majority vote of its neighbors. A case being assigned to the class most common amongst its k-nearest neighbors measured by a distance function. In case of k = 1, it is simply assigned to the class of its nearest neighbor. The value of k can be anywhere between 1 and the total number of instances in the training data set. Lower value of k might result in a noisier result. So, in general, a large value of k is more

precise as it generally reduces the overall noise but there is no guarantee. Cross-validation is also an effective way to determine an optimal value of k by using an independent data set to validate the k value. Now, if k is equal to the total number of instances in the training data set, for any query instance, all the other instances in the training set becomes the nearest neighbor. If this happens, the predicted response for a new instance is just the frequent response variable in the training set. This is comparatively simpler compared to other machine learning algorithm and produces predictions with high accuracy. With a large value of k when the number of the training instances becomes very big, it has been observed the asymptotic error rates gets aligned with the optimal Bayes error rate. For this reason, the KNN mechanism has a standard comparison method against which any new classifiers, such as neural networks, can be compared.

With all the advantages of KNN, however, it has been criticized with the assumptions that (i) all data $(x, f(x))$ must be stored in memory at the same time to implement such a rule; (ii) the nearest neighbor rules are not well suited for fast parallel computation. While it is true that in KNN the testing phase may take a significant amount of time but with the exponential progress in computation power and cheaper hardware, this is not a huge problem anymore [22-25].

3.2.2 SVM classifier

Support vector machine (SVM) technique was first proposed by Vapnik. It is a statistical learning theory designed to solve problems related to classification, learning and regression. Compared to other classifiers, SVM implements the principle of structural risk minimization, it evades local minima, solve issues like over fitting and can provide better generalization. To draw boundaries between the classes, it establishes a hyperplane or a set of hyperplanes in a high dimension feature space. For classification there can be multiple hyperplanes for separation but the best hyperplane produces maximum margin between the data points of two classes. In many cases the data points are not linearly separable in the input space so they need nonlinear transformation into a high dimensional space and then the linear maximum margin classifier can be applied. Kernel functions are used for this purpose. Selection of an appropriate kernel for a certain classification problem influence the performance of the SVM because different kernel function constructs different SVMs and affects the generalization ability and learning ability of

SVM. There is no theoretical method for selecting kernel function and its parameters. Presently Gaussian kernel is the kernel function which is mostly used because of its good features [26-28].

In the following, scalars are denoted with italic lowercases (e. g. , y, b), vectors with bold lowercases (e. g. , w, x), and matrices with italic uppercases (e. g. , W). w^T is the transpose of w, and $\| w \| = w^T * w$.

Let,

x be a feature vector (i.e., the input of the SVM). $x \in R^n$, where n is the dimension of the feature vector.
y be the class (i.e., the output of the SVM). $y \in \{-1,1\}$, i.e. the classification task is binary.
w and b be the parameters of the SVM: we need to learn them using the training set.$(x(i), y(i))$ be the ith sample in the data set. Let's assume we have N samples in the training set.

With $n = 2$, one can represent the SVM's decision boundaries as follows:

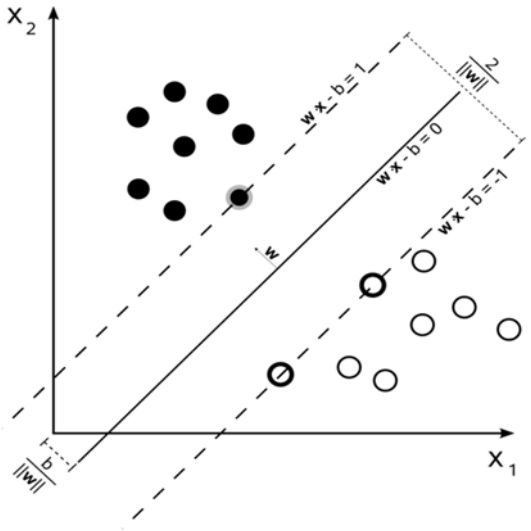


Figure 7: SVM hyperplane.

The class y is determined as follows:

$$y(i) = \begin{cases} -1, & if\ w^T x^{(i)} + b \leq -1 \\ 1, & if\ w^T x^{(i)} + b \geq 1 \end{cases} \tag{22}$$

which can be more concisely written as

$$y(i)(w^T x(i) + b) \geq 1 \tag{23}$$

Goals of SVM

The SVM aims at satisfying two requirements:

- The SVM should maximize the distance between the two decision boundaries. Mathematically, this means we want to maximize the distance between the hyperplane defined by $w^T x + b = -1$ and the hyperplane defined by $w^T x + b = 1$. This distance is equal to $\frac{2}{\|w\|}$. This means we want to solve $\max_{w} \frac{2}{\|w\|}$. Equivalently we want to solve $\min_{w} \frac{\|w\|}{2}$.

- The SVM should also correctly classify all $x^{(i)}$, which means

$$y(i)\left(w^T x^{(i)} + b\right) \geq 1, \qquad where, i \in \{1, \dots, N\} \tag{24}$$

This leads us to the following quadratic optimization problem:

$$\min_{w,b} \frac{||w||}{2}$$

$$s.t. \quad y(i)\left(w^T x^{(i)} + b\right) \geq 1, \qquad where, i \in \{1, \dots, N\} \tag{25}$$

This is the hard-margin SVM, as this quadratic optimization problem admits a solution if the data is linearly separable.

One can relax the constraints by introducing so-called slack variables $\xi^{(i)}$. Note that each sample of the training set has its own slack variable. This gives us the following quadratic optimization problem:

$$\min_{w,b} \frac{\| w \|}{2} + C \sum_{i=1}^{N} \xi^{(i)} \tag{26}$$

$$s.t. \, y^{(i)}\left(w^T x^{(i)} + b \geq 1 - \xi^{(i)}\right), \quad where, i \in \{1, \dots, N\} \tag{27}$$

$$\xi^{(i)} \geq 0, \quad where, i \in \{1, \dots, N\} \tag{28}$$

This is the soft-margin SVM. C is a hyperparameter called penalty of the error term. (One can add even more flexibility by introducing a function ϕ that maps the original feature space to a higher dimensional feature space. This allows non-linear decision boundaries. The quadratic optimization problem becomes:

$$\min_{w,b} \frac{\| w \|}{2} + C \sum_{i=1}^{N} \xi^{(i)} \tag{29}$$

$$s.t. \, y^{(i)}\left(w^T \phi(x^{(i)}) + b \geq 1 - \xi^{(i)}\right), \quad where, i \in \{1, \dots, N\} \tag{30}$$

$$\xi^{(i)} \geq 0, \quad where, i \in \{1, \dots, N\} \tag{31}$$

Optimization Procedure

The quadratic optimization problem can be transformed into another optimization problem named the Lagrangian dual problem (the previous problem is called the primal):

$$\max_{\alpha} \min_{w,b} \frac{\| w \|}{2} + C \sum_{i=1}^{N} \alpha^{(i)}(1 - w^T \phi(x^{(i)}) + b) \tag{32}$$

$$s.t \;\; 0 \leq \alpha^{(i)} \leq C \, , \quad \forall i \in \{1, \dots, N\} \tag{33}$$

Making a Predication

Once the α(i) are learned, one can predict the class of a new sample with the feature vector xtest as follows:

$$y_{test} = sign(w^T \phi(x_{test}) + b) \quad (34)$$

$$y_{test} = sign(\sum_{i=1}^{N} \alpha^{(i)} y^{(i)} \phi(x^{(i)}) T \phi(x_{test}) + b) \quad (35)$$

The summation means one must sum over all the training samples, but the clear majority of $\alpha(i)$ are 0. So, in practice it isn't an issue. (note that one can construct special cases where all $\alpha(i) > 0$.) $\alpha(i) = 0$ if $x(i)$ is a support vector. The illustration above has 3 support vectors.

Kernel trick

One can observe that the optimization problem uses the $\phi(x(i))$ only in the inner product $\phi(x(i))^T \phi(x^{(j)})$. The function that maps $(x^{(i)}, x^{(j)})$ to the inner product $\phi(x^{(i)})^T \phi(x^{(j)})$ is called a kernel, a.k.a. kernel function, often denoted by k.

One can choose k so that the inner product is efficient to compute. This allows to use a potentially high feature space at a low computational cost. That is called the kernel trick. For a kernel function to be valid, i.e. usable with the kernel trick, it should satisfy two key properties. There exist many kernel functions to choose from. As a side note, the kernel trick may be applied to other machine learning models, in which case they are referred as kernelized.

3.2.3 Random Forest

The term CART (Classification and Regression Trees) was first coined by Leo Breiman in his work on random forest in 2001. Decision tree algorithm can be used both for classification and regression. Here in our thesis, we will be focusing on the classification aspect of random forest algorithm. The aim at each stage is to associate desired output values with specific values of a variable. The result is a decision-tree in which each path identifies a combination of values associated with a prediction. Random forest is an ensemble technique. Ensembles are a divide-and-conquer approach used to improve performance. The main principle behind ensemble

methods is that a group of “weak learners” can come together to form a “strong learner”. The figure below (taken from here) provides an example. Each classifier, individually, is a “weak learner,” while all the classifiers taken together are a “strong learner”.

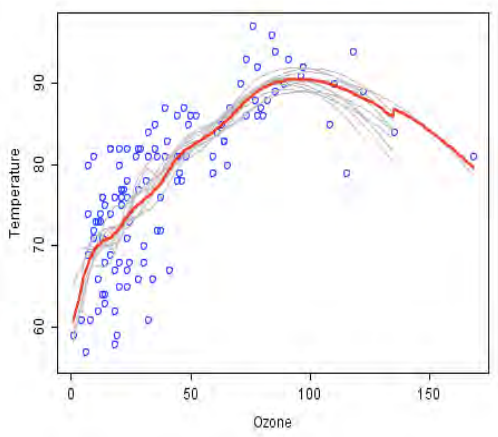


Figure 8: Multiple weak learning machines give out a strong prediction.

The grey lines in the figure show the predictions of multiple weak learning machines. Although their individual prediction is not quite accurate. Their average or weighted average prediction can come out as a strong predictor. The red line shows the final strong outcome.

Random Tree Structure

There are mainly three ways a random decision-making tree can be generated. These are

- The method for splitting the leaves (leaf) in a tree
- The type of predictor to use in each leaf
- The method of creating randomness into the prediction mechanism

Each non-leaf node in this tree is basically a decision maker. These nodes are called decision nodes. Each node carries out a specific test to determine where to go next. Depending on the outcome, you either go to the left branch or the right branch of this node. We keep doing this until we reach a leaf node. If we are constructing a classifier, each leaf node represents a class. Let’s say we are trying to determine whether or not it’s going to rain tomorrow based on three factors available today — temperature, pressure, and wind. So, a typical decision tree would look like this:

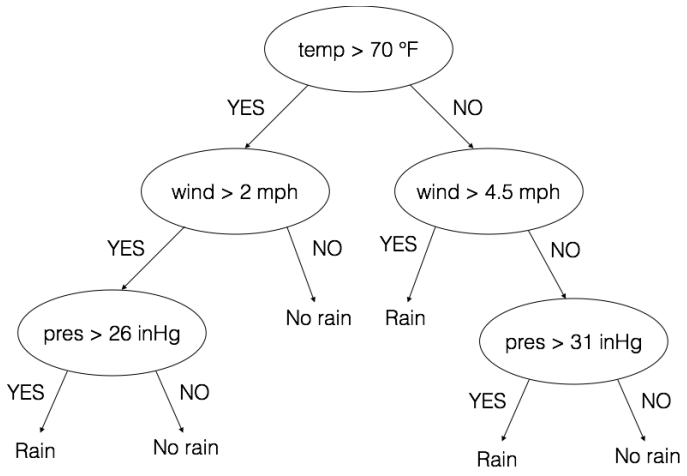


Figure 9: Decision Tree Generation.

To split a leaf, a collection of candidate splits is generated and a criterion is evaluated to choose between them. A simple strategy is to choose among the candidates uniformly at random. A more common approach is to choose the candidate split which optimizes a purity function over the leaves (leaf) that would be created. A typical choice here is to maximize the information gain. The most common choice for predictors in each leaf is to use the average response over the training points which fall in that leaf. We consider only simple averaging predictors here.

Injecting randomness into the tree construction can happen in many ways. The choice of which dimensions to use as split candidates at each leaf can be randomized, as well as the choice of coefficients for random combinations of features. In either case, thresholds can be chosen either randomly or by optimization over some or all the data in the leaf. Another common method for introducing randomness is to build each tree using a bootstrapped or sub-sampled data set. In this way, each tree in the forest is trained on slightly different data, which introduces differences between the trees.

Splitting Leaves and generating Random Forest

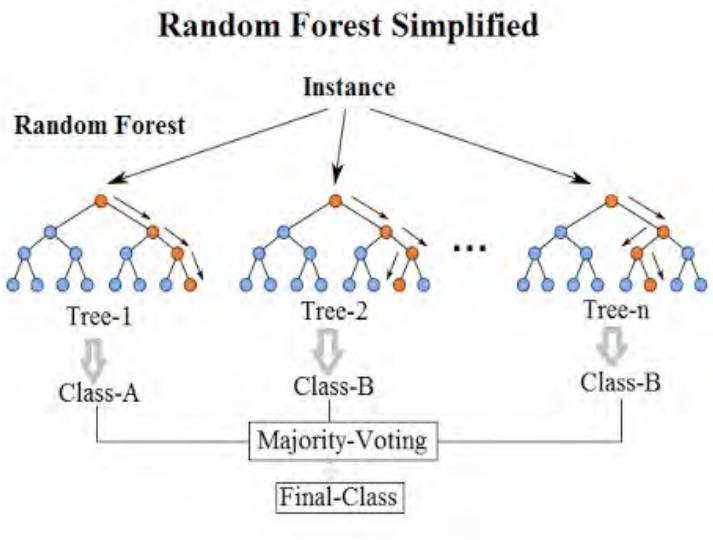


Figure 10: Simplified Leaf Generation Process in Random Forest Algorithm [30]

In order to construct the optimal tree, we need to know what attribute should be at the root node and what should go to the leaf node. Gini index is used as the cost function to evaluate the splits in the dataset. The aim is to minimize the index.

$$Gini(t) = 1 - \sum_{i=0}^{c-1} [P(i|t)]^2 \tag{36}$$

Where,

$$c = Number\ of\ classes$$
$$i = Weight\ of\ splitted\ leaf$$
$$t = Total\ weight\ of\ the\ parent\ leaf$$

A split in the dataset involves one input attribute and one value for that attribute. It can be used to divide training patterns into two groups of rows. A Gini score gives an idea of how good a split is by how mixed the classes are in the two groups created by the split. A perfect separation results in a Gini score of 0, whereas the worst case split that results in 50/50 classes in each group results in a Gini score of 1.0 (for a 2 class problem). A Gini score gives an idea of how good a split is by how mixed the classes are in the two groups created by the split. A perfect separation results in a Gini score of 0, whereas the worst case split that result in 50/50 classes.

We calculate it for every row and split the data accordingly in our binary tree. We repeat this process recursively.

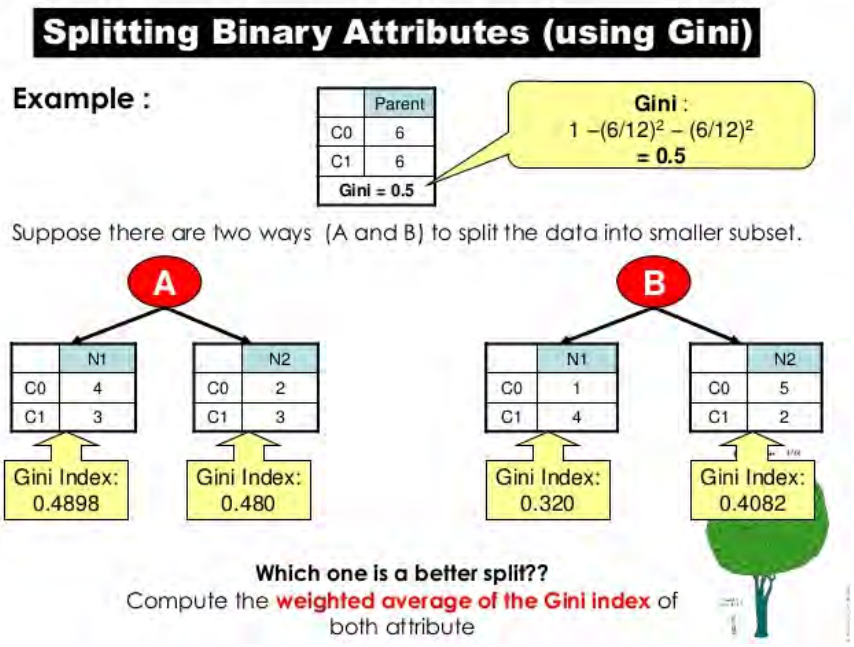


Figure 11: Leaf creation using Gini index [30]

To summarize the process:

- Sample N cases at random with replacement to create a subset of the data (see top layer of figure below). The subset should be about 66% of the total set.
- At each node:

  1. For some number m (see below), m predictor variables are selected at random from all the predictor variables.

  2. The predictor variable that provides the best split, according to some objective function, is used to do a binary split on that node. Here Gini index is used as the cost function.

  3. At the next node, choose another m variables at random from all predictor variables and do the same. This is done recursively until all the variables are exhausted.

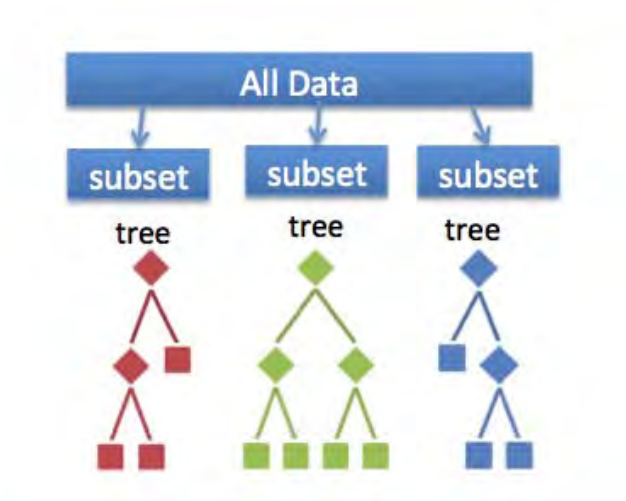


Figure 12: Random subset selection and tree generation.

Properties

- With a large number of predictors, the eligible predictor set will be quite different from node to node.
- The greater the inter-tree correlation, the greater the random forest error rate, so one pressure on the model is to have the trees as uncorrelated as possible.
- As m goes down, both inter-tree correlation and the strength of individual trees go down. So, some optimal value of m must be discovered.

Strength and Weakness

Random forest runtimes are quite fast, and they are able to deal with unbalanced and missing data. Random Forest weaknesses are that when used for regression they cannot predict beyond the range in the training data, and that they may over-fit data sets that are particularly noisy [29-32].

3.2.4 Naïve Bayes

Naïve Bayes is a classification technique that depends on Bayes Theorem. This algorithm assumes independence among the predictors. It assumes that the presence of a particular feature in a specific class is unrelated to the presence of any other feature. In our system, this means that we are assuming that a feature of a sample is not dependent on any other features in that sample.

This assumption is however, not always accurate and hence, this method is called a naïve method. Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability $P(c|x)$ from $P(c), P(x)$ and $P(x|c)$. Look at the equation below:

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \tag{37}$$

- P(c|x) is the posterior probability of class (target) given predictor (attribute).
- P(c) is the prior probability of class.
- P(x|c) is the likelihood which is the probability of predictor given class.
- P(x) is the prior probability of predictor (attribute).

Advantages of Naïve Bayes

- It is easy and fast to predict class of test data set. It also performs well in multi class prediction
- When assumption of independence holds, a Naive Bayes classifier performs better compare to other models like logistic regression and less training data is needed.
- It performs well in case of categorical input variables compared to numerical variable(s). For numerical variable, normal distribution is assumed (bell curve, which is a strong assumption).

Limitations of Naïve Bayes

- If categorical variable has a category (in test data set), which was not observed in training data set, then model will assign a 0(zero) probability and will be unable to make a prediction. This is often known as "Zero Frequency". To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation.

- On the other side naive Bayes is also known as a bad estimator, so the probability outputs from predicted probability machines are not to be taken too seriously.
- Another limitation of Naive Bayes is the assumption of independent predictors. In real life, it is almost impossible that we get a set of predictors which are completely independent.

Types of Naïve Bayes

***Gaussian*:** It is used in classification and it assumes that features follow a normal distribution.

Multinomial: It is used for discrete counts.

We used Gaussian Naïve Bayes for our data analysis

3.2.5 Artificial Neural Network

A multilayer Perceptron is an ANN that consists of multiple layers of neurons in a feed forward architecture. It is a variant of the original Perceptron model first proposed by Rosenblatt in 1950. An MLP consists of three or more layers- an input layer, an output layer and a single or multiple hidden layer connecting the input and the output layers. The direction of these layers always flows from lower layers to upper layers and the neurons in the same layer do not have any connection between them. see Fig.1. Generally, the number of neurons in the input layer is equal to the number of working attributes for a pattern recognition problem and the neurons number in the output layer is equal to the number of classes.

MLP passes the processed data of each neuron through a nonlinear activation function and every neuron of a layer is fully connected to the next layer. The perceptrons help to create decision boundaries between patterns. Each perceptron provides a non-linear mapping to a new dimension. Since in MLP each network is completely connected with the next network, each neuron in a layer is connected to the next layer with a weight function w_{ij}. MLP uses a supervised learning technique called back propagation. During the training, the weight function is defined and adjusted so that multiplying the weight matrix with the output of the last hidden layer gives us the desired class output. MLP trains data by minimizing the chosen cost function until it reaches a specific value. This was originally developed by Werbos and Parker [33].

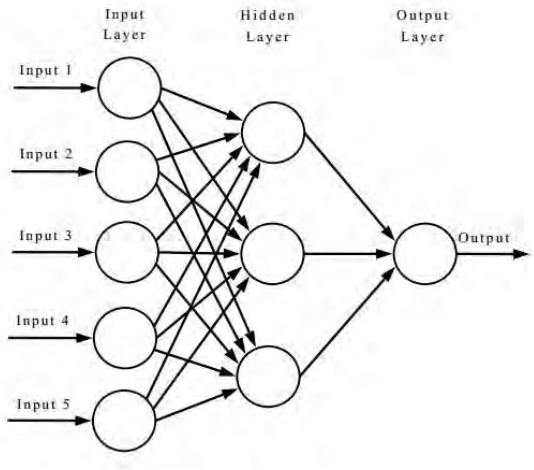


Figure 13: Basic architecture of a 3- layer feed-forward network.

Assuming we used an input layer where there are N attributes. So the number of input neurons will also be N. Let us say, we used Relu as the activation function in each layer. So, the input array,

$$X = [x_0, x_1, x_2, \dots, x_N]^T \tag{38}$$

Activation function,

$$\sigma(x) = \max(x, 0) \tag{39}$$

Fig 3.14. is an example of an MLP network, where a hidden layer consists of M neurons and each neuron has N weights, which can be presented in a $M \times N$ matrix called Input Weight matrix $I_w$as shown in equation40. The input vector X is multiplied by I_w and the resulting matrix is summed with a bias vector $b1$ to form the vector $n1$ as shown in equation 37. The output of the hidden layer $h1$ is the result of applying the activation function on $n1$ (see equation 38).

$$I_w = \begin{bmatrix} w_{1,1} w_{1,2} & \dots w_{1,N} \\ w_{2,1} w_{2,2} & \dots w_{2,N} \\ \dots & \\ w_{M,1} w_{M,2} & \dots w_{M,N} \end{bmatrix} \tag{40}$$

$$n_1 = I_w . X + b_1 \tag{41}$$

$$h_1 = \sigma(n_1) \tag{42}$$

Applying the same operation on the hidden layer will give us the output layer which consists of K neurons and here $h1$ is used as the input vector (equation 39, 40 and 41).

$$L_w = \begin{bmatrix} w_{1,1} w_{1,2} & \dots w_{1,M} \\ w_{2,1} w_{2,2} & \dots w_{2,M} \\ \dots & \\ w_{K,1} w_{K,2} & \dots w_{K,M} \end{bmatrix} \tag{43}$$

$$n_2 = L_w . h_1 + B_2 \tag{44}$$

$$h_2 = \sigma(n_2) \tag{45}$$

Chapter 04

Proposed Method of Facial Expression Detection

4.1 Feature Sets and Emotion

In order to train machines to identify and classify different emotions, first, we need to devise ways of quantifying the emotions from 2D pictures. Since we used single modal information, we only had to train and classify the algorithms with the black and white 2D image of facial expressions. For this purpose, we used the JAFFE dataset.

4.1.1 Dataset Description

The database contains 213 images of 7 facial expressions (6 basic facial expressions + 1 neutral) posed by 10 Japanese female models. The emotions are anger, disgust, happiness, neutral, sadness, surprise. Each image has been rated on 6 emotion adjectives by 60 Japanese subjects. All the images are black and white, in Tiff format and have 256 x 256 resolutions. The database was planned and assembled by Michael Lyons, Miyuki Kamachi, and Jiro Gyoba. We thank Reiko Kubota for her help as a research assistant. The photos were taken at the Psychology Department in Kyushu University [34].

Figure 14: Sample photos from the JAFFE dataset.

4.2 Method 1: Classification Algorithms on Unprocessed Raw Dataset

To demonstrate how different machine learning algorithms can perform the classification of emotion, we have taken two different approaches. First, we took the images, extracted the raw pixel values and performed data decomposition to plot the higher dimensional data in a lower dimensional feature space. We assessed from that how classifiable our dataset is. We visualized the probability density function of

PCA and t-SNE components. Then we applied different machine learning algorithms using Python based Scikit-learn libraries. We applied SVM (kernel: linear, radial basis function and polynomial), adaboost, decision tree, random forest, extra tree, KNN, logistic regression classifier and Gaussian Naïve Bayes algorithm on the dataset to observe and compare that how each of the algorithm can classify trained emotion as well as how well they generalize on newer dataset.

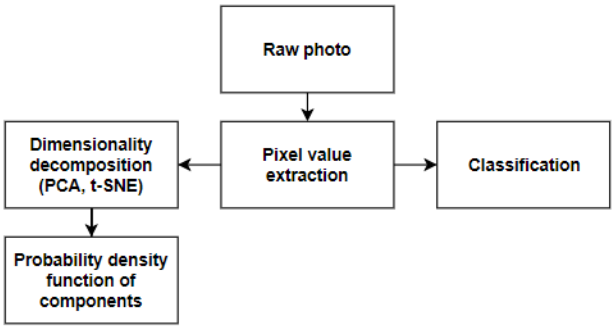


Figure 15: Workflow of method 1.

4.3 Method 2: Classification Algorithms on Deliberately Extracted Feature Sets

In the second approach we used Python based "face_recognition" algorithm to extract deliberate points on the faces to make the classification process easier for the algorithms. Each facial image data was divided into 9sub-regions. The sub-regions were i) chin ii) left eyebrow iii)right eyebrow iv) nose bridge v) nose tip vi) left eye vii) right eye viii) top lip ix) bottom lip. We got 72 face landmarks for each sample. We created a 1x144 feature vector for each sample [35].

Figure 16: Extracted feature regions.

We applied PCA and t-SNE on the feature matrix and observed the probability density function of the components. Then we applied the same classifiers we used on the first method and compared the results. Finally, we applied neural network based classifier on the extracted feature sets to stack it against other classifier applied both on raw dataset and extracted feature sets.

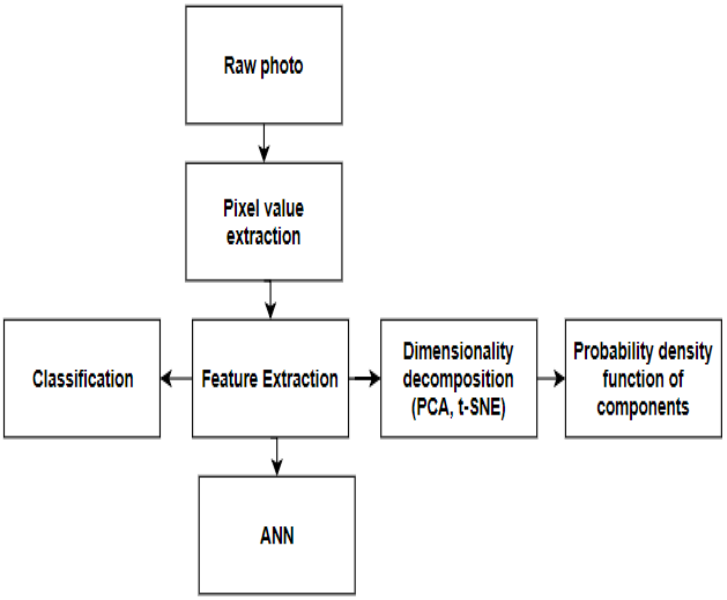


Figure 17: Workflow of method 2.

Chapter 05

Setup and Results

5.1 Experimental Setup

We used a pc with 2.7ghz core i-5 6200u processor, 12 gigabyte ddr3 1600mhz memory, NVIDIA 940MX GPU with 2 gigabytes graphics memory for running our algorithms. Python based libraries were used for applying the data decomposition and constructing the classification algorithms.

- Numpy library: For manipulating the data arrays
- Matplotlib: For plotting the attributes
- OpenCV: For photo manipulation
- Seaborn: For plotting data decomposition
- Scikit-learn: For constructing and applying the classifier

5.2 Method 1 Results

First, we applied data decomposition techniques on the raw images and plotted them on two-dimensional feature space. We encoded the emotions as follows:

- Anger-0
- Disgust-1
- Fear-2
- Happiness-3
- Neutral-4
- Sadness-5
- Surprised-6

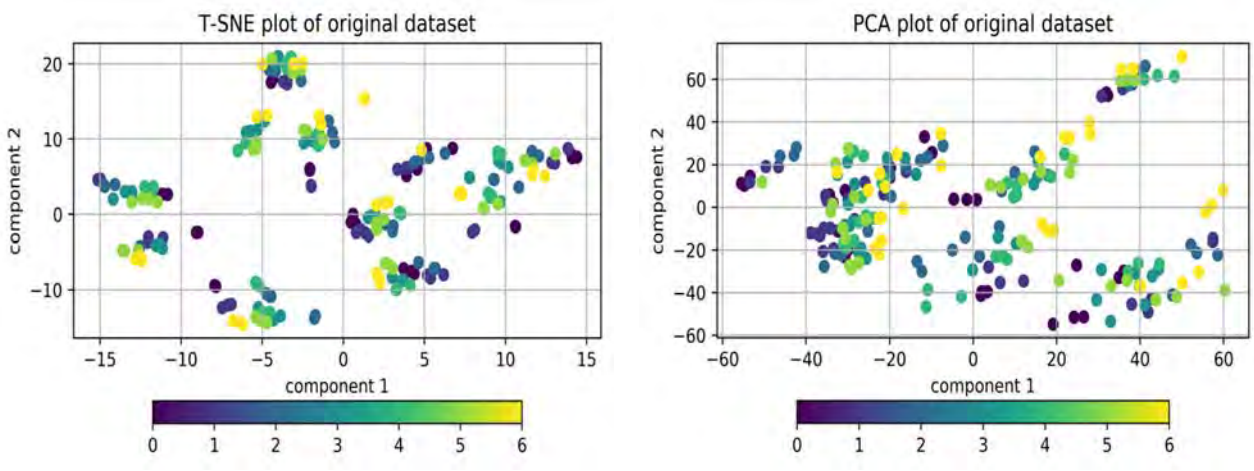


Figure 18: Data decomposition of raw image data.

The decomposed components of the raw data show that the algorithms could not exactly clusterize the emotions. However, since PCA only takes the variance of the attributes into account, it did a poorer job compared to the probabilistic method of t-SNE.

We also plotted the probability density function of the component of PCA and t-SNE to visualize the distribution of the components. This allows us to select a better pre-processing method based on data distribution. Better preprocessing often yields to better classification results.

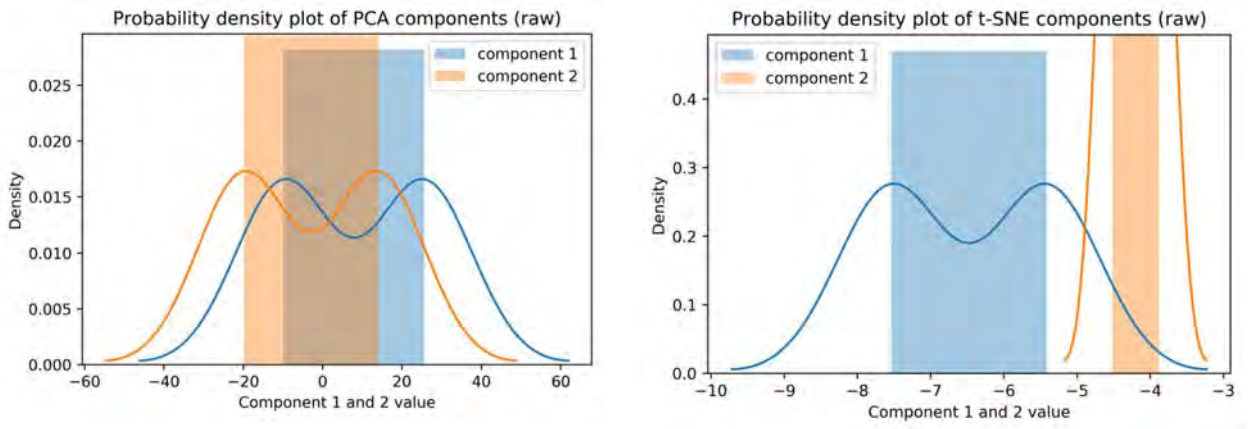


Figure 19: PDF of decomposed components (extracted features).

In both cases the first component has bi modal distribution but t-SNE's second component has gaussian bell shaped distribution and the peak is much larger than the first component's peak. Moreover, both the components in every case are overlapping each other which may denote the fact that the classifiers will have difficulties to discern the emotions from one another. We normalized the dataset by dividing each pixel value with the maximum pixel value (255).

After performing data decomposition, we applied different machine learning classifiers to classify the emotions. We also kept the record of how long the classifiers take to perform the job. The results are shown in table 1.

Table 1: Classifier performance on Raw data

Classifier	*Accuracy*	*Runtime*
SVM (kernel: linear)	85.75	6.54s
SVM (kernel: RBF)	50	6.99s
SVM (kernel: polynomial)	37.5	6.67s
Adaboost	15.625	11.75s
Decision Tree	46.875	7.05s
Random Forest	59.375	0.59s
Extra Tree	71.875	6.99s
KNN	15.625	2.8s
logR	90.625	72.72s
Naïve Bayes	37.5	1.20

5.3 Method 2 Results

We applied the "face_recognition" algorithm to extract relevant features from each image and instead of applying the machine learning algorithms directly on the images we applied them on the deliberately extracted feature sets. Each photo corresponds to a 1 x 144 feature vector. So, our feature matrix for 213 photos were 213 x 144. We applied the same decomposition algorithms PCA, t-SNE so visualize how well the individual emotions were clusterized. Then we

again applied the classifiers used on method 1 and compared how they performed against each other across different methods.

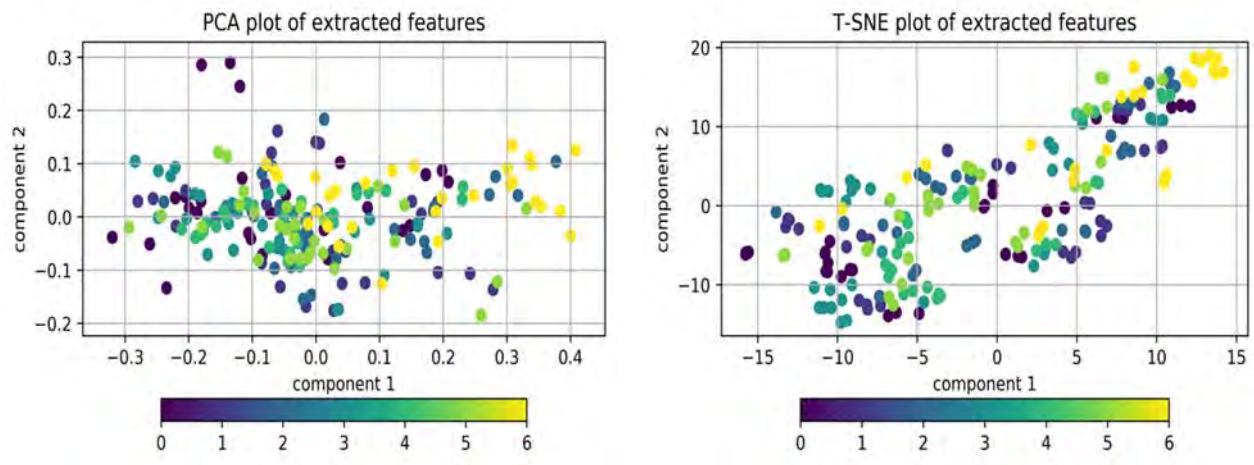


Figure 20: Data decomposition of extracted features.

The algorithms performed noticeably better on the extracted features compared to the raw pixel values. Overlapping classes are far less than that of the first method.

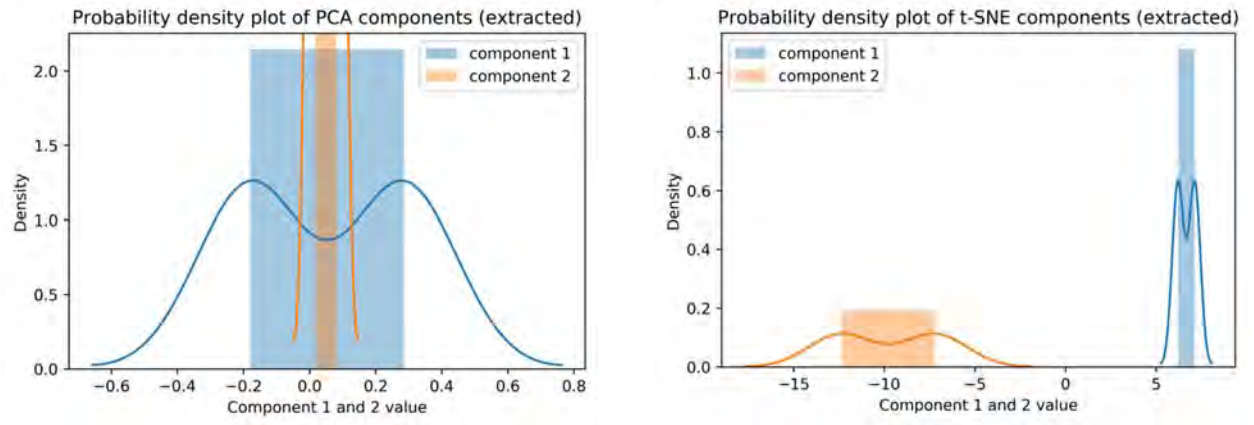


Figure 21: PDF of decomposed components (extracted features).

Here in case of t-SNE the PDF's of the two components are not overlapping each other and this indicates that the classifiers will perform better discerning the emotions from one another.

Similar to the first method we also applied and recorded the performance of different classification algorithms on the extracted feature matrix.

Table 2: Classifier performance on extracted feature dataset

Classifier	*Accuracy*	*Runtime*
SVM (kernel: linear)	84.375	15.62 ms
SVM (kernel: RBF)	78.125	15.62 ms
SVM (kernel: polynomial)	43.750	15.62 ms
Adaboost	18.750	36.68 ms
Decision Tree	65.625	15.62 ms
Random Forest	62.500	31.25 ms
Extra Tree	68.750	15.62 ms
KNN	40.625	64.41 ms
logR	84.375	15.62 ms
Naïve Bayes	56.250	31.21 ms

Comparing table 1 and 2, it is evident that the method 2 results outperforms method results by a large margin. Almost all the classifiers yielded better results when applied on the extracted features. The PDF of the second method also forecasted that. So, it can safely be said, that extracting deliberate features can be really helpful in case of performing classification of facial expression. We also applied vanilla neural network on the extracted feature matrix to see how the model converges.

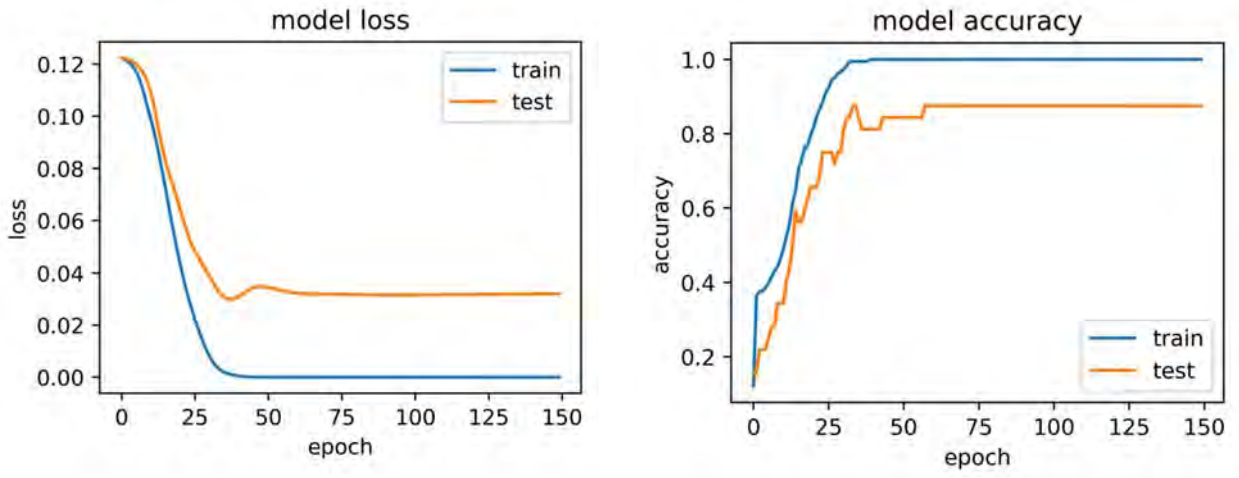


Figure 22*: Accuracy and loss of ANN on the extracted feature set*

Our ANN model achieved an impressive 90.63% validation accuracy on the testing dataset.

Chapter 06

Conclusion

The main purpose of working with this model was to explore in the innovative world of facial expression. While at the same time working with this model we utilized the element extraction strategy proposed in Principal segment investigation (PCA) and t-disseminated Stochastic Neighbor Embedding (t-SNE) for dimensionality diminishment and visual portrayal are required element for segments in a 2D include space. Different classification algorithms like K-nearest Neighbor (KNN), Support Vector Machines (SVM), Gaussian Naïve Bayes, Random Forest, Extra Tree, Ensemble machines and vanilla neural networks have been widely used for classification and recognition tasks. We compared the performance of different classifiers on Japanese Female Facial Expression (JAFFE) database for 80 % training and 20 % testing. Before and after performing deliberate feature extraction procedure by Artificial Neural Network which gave us a accuracy rate of 90.63%. During this procedure we faced few limitations as we worked with all the pixel data in the first method it can be regarded as somewhat crude. However, our second method of feature extraction achieves better accuracy across all the classifiers. In bigger dataset with higher classes, often the accuracy dips below and acceptable level which is another limitation faced by us. These algorithms often fail to grasp the inner patter of the features and classify accordingly. To solve this, Convolutional Neural Network based models can be used to improve accuracy on bigger and more complicated dataset. This work shows that deliberate feature extraction improves model accuracy as well as runtime. In the future work field, with bigger and more complicated dataset, our work can be made more robust and impervious to noise and other factors. We showed how different machine learning algorithms performs against each other in validation accuracy and we aimed at building a system that not only can discern between different emotions using only single modal information but also will be able to generalize on dataset that the system has never encountered before.

References

[1] Bartlett, M.S., Hager, J.C., Ekman, P., Sejnowski, T.J., 1999. Measuring facial expressions by computer image analysis. Psychophysiology 36, 253–263.

[2] Lyons, M., Akamatsu, S., Kamachi, M., Gyoba, J., 1998. Coding facial expressions with Gabor wavelets. In: Proceedings of the Third IEEE International Conference on Automatic Face and Gesture Recognition.

[3] Friudlund, A.J. What do facial expressions express? /http://www.sscnet. ucla.edu/anthro/bec/papers/Fridlund_Facial_Expressions.PDFS (Visited November 2006).

[4] Elfenbein, H.A., Ambady, N., 2002. On the universality and cultural specificity of emotion recognition: a meta-analysis. Psychological Bulletin 128 (2), 205–235.

[5] Elfenbein, H.A., Ambady, N., 2003. When familiarity breeds accuracy: cultural exposure and facial emotion recognition. Journal of Personality and Social Psychology 85 (2), 276–290.

[6] Cohn, J., Kanade, T., 2006. Use of automated facial image analysis for measurement of emotion expression. In: Coan, J.A., Allen, J.B. (Eds.), The Handbook of Emotion Elicitation and Assessment. Oxford University Press Series in Affective Science

[7] Fasel, I.R., Bartlett, M.S., Movellan, J.R., 2002. A comparison of Gabor filter methods for automatic detection of facial landmarks. In: Proceedings of the 5th International Conference on Face and Gesture Recognition

[8] Zhang, Z., Lyons, M., Schuster, M., Akamatsu, S., 1998. Comparison between geometry-based and Gabor wavelets-based facial expression recognition using multi-layer perceptron. In: Proceedings of the Third IEEE International Conference on Automatic Face and Gesture Recognition.

[9] Ageitgey, “ageitgey/face_recognition,” GitHub, 09-Jul-2018. [Online]. Available: https://github.com/ageitgey/face_recognition. [Accessed: 12-Jul-2018].

[10] De Stefano, C., Sansone, C., Vento, M., 1995. Comparing generalization and recognition capability of learning vector quantization and multilayer perceptron

architectures. In: Proceedings of the 9th Scandinavian Conference on Image Analysis, June, pp. 1123–1130

[11] Rani, Meesala Shobha; S, Sumathy (26 September 2017). "Perspectives of the performance metrics in lexicon and hybrid based approaches: a review". International Journal of Engineering & Technology. 6 (4): 108. doi:10.14419/ijet.v6i4.8295

[12] Cambria, Erik; Poria, Soujanya; Bajpai, Rajiv; Schuller, Bjoern (2016). "SenticNet 4: A Semantic Resource for Sentiment Analysis Based on Conceptual Primitives". Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers.

[13] Balahur, Alexandra; Hermida, JesúS M.; Montoyo, AndréS (1 November 2012). "Detecting implicit expressions of emotion in text: A comparative analysis". Decision Support Systems. 53 (4): 742–753. doi:10.1016/j.dss.2012.05.024. ISSN 0167-9236

[14] Hemmatian, Fatemeh; Sohrabi, Mohammad Karim (18 December 2017). "A survey on classification techniques for opinion mining and sentiment analysis". Artificial Intelligence Review. doi:10.1007/s10462-017-9599-6.

[15] Majumder, Navonil; Poria, Soujanya; Gelbukh, Alexander; Cambria, Erik (March 2017). "Deep Learning-Based Document Modeling for Personality Detection from Text". IEEE Intelligent Systems. 32 (2): 74–79. doi:10.1109/MIS.2017.23

[16] Mahendhiran, P. D.; Kannimuthu, S. (May 2018). "Deep Learning Techniques for Polarity Classification in Multimodal Sentiment Analysis". International Journal of Information Technology & Decision Making. 17 (03): 883–910. doi:10.1142/S0219622018500128

[17] Yu, Hongliang; Gui, Liangke; Madaio, Michael; Ogan, Amy; Cassell, Justine; Morency, Louis-Philippe (23 October 2017). "Temporally Selective Attention Model for Social and Affective State Recognition in Multimedia Content". ACM: 1743–1751. doi:10.1145/3123266.3123413

[18] 10154969000202429, “A One-Stop Shop for Principal Component Analysis – Towards Data Science,” Towards Data Science, 17-Apr-2017. [Online]. Available: https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c. [Accessed: 12-Jul-2018].

[19]

[20] Roweis,S.T,Saul,L.k.,2000.”Nonlinear Dimensionality Reduction by Locally Linear

Embedding",Science,290(5500),2323-2326 DOI:10.1126/science.290.5500.2323

[21] Free PowerPoint Templates. (2018). Free Machine learning diagram - Free PowerPoint Templates. [online] Available at: https://yourfreetemplates.com/free-machine-learning-diagram [Accessed 2 Aug. 2018].

Opensourcefaisal.blogspot.com. (2018). Implementasi Algoritma Nearest Neighbor | BerbagiIlmu. [online] Available at: http://opensourcefaisal.blogspot.com/2016/04/implementasi-algoritma-nearest-neighbor.html [Accessed 2 Aug. 2018].

[22] Devroye L, Gyori L, Lugosi G. A Probabilistic Theory of Pattern Recognition. New York: Springer-Verlag, 1996

[23] Dasarathy B V. Nearest neighbor norms-NN pattern classification techniques. Los Alamitos: IEEE Comp, Society Press, 1991

[24] Vladimir, V.N.: The Nature of Statistical Learning Theory. Springer, Heidelberg (1995)

[25] Xiao, H., Peng, F., Wang, L., Li, H.: Ad hoc-based feature selection and support vector machine classifier for intrusion detection. In: IEEE International Conference on Grey Systems and Intelligent Services (GSIS 2007), pp. 1117–1121 (2007) Gao, M., Tian, J., Xia, M.: Intrusion Detection Method Based on Classify Support Vector Machine. In: Proceedings of the 2009 Second International Conference on Intelligent Computation Technology and Automation, pp. 391–394 (2009)

[26] Ahmad, I., Abdulah, A., Alghamdi, A.: Towards the Designing of a Robust Intrusion Detection System through an Optimized Advancement of Neural Networks. In: Kim, T.-h., Adeli, H. (eds.) AST/UCMA/ISA/ACN 2010. LNCS, vol. 6059, pp. 597–602. Springer, Heidelberg (2010)

[27] Yang, M.-h., Wang, R.-c.: DDoS detection based on wavelet kernel support vector machine. The Journal of China Universities of Posts and Telecommunications 15(3), 59–63, 94 (2008)

[28] Kumar, G., Kumar, K., Sachdeva, M.: The use of artificial intelligence based techniques for intrusion detection: a review. Artificial Intelligence Review 34(4), 369–387 (2010)

[29] Breiman, L. Bagging predictors. Machine Learning, 24(2):123–140, 1996.

[30] Breiman, L. Random forests. Machine Learning, 45(1):5–32, 2001.

[31] Breiman, L. Consistency for a simple model of random forests.Technical report, University of California at Berkeley, 2004.

[32] Breiman, L., Friedman, J., Stone, C., and Olshen, R. Classification and Regression Trees. CRC Press LLC, 1984

[33] N. Buduma and N. Locascio, Fundamentals of deep learning: designing next-generation machine intelligence algorithms. Sebastopol, CA: OReilly Media, 2017.

[34] Facial Expression Database: Japanese Female Facial Expression (JAFFE) Database. [Online]. Available: http://www.kasrl.org/jaffe.html. [Accessed: 13-Jul-2018].

[35] Ageitgey, "ageitgey/face_recognition," GitHub, 09-Jul-2018. [Online]. Available: https://github.com/ageitgey/face_recognition. [Accessed: 12-Jul-2018].