An Efficient Approach to Real-Time Face Recognition From Digital images.

A Project Report

Submitted By

MD. Raihanul Kabir

ID No- 1608347

Session- 2020

DEPARTMENT OF STATISTICS

Hajee Mohammad Danesh Science And Technology University Dinajpur-5200

An Efficient Approach to Real-Time Face Recognition From Digital images.

A Project Report

Submitted By

MD. Raihanul Kabir

ID No- 1608347

Submitted to

A.S.M Abu Saeed	**Rajib Dey**
(Assistant Professor)	**(Associate Professor)**
SUPERVISOR	**CO-SUPERVISOR**

Prof. Dr. Md. Earfan Ali Khondaker

CHAIRMAN OF EXAMINATION COMMITTEE

DEPARTMENT OF STATISTICS

Hajee Mohammad Danesh Science And Technology University Dinajpur-5200

CHAPTER ONE

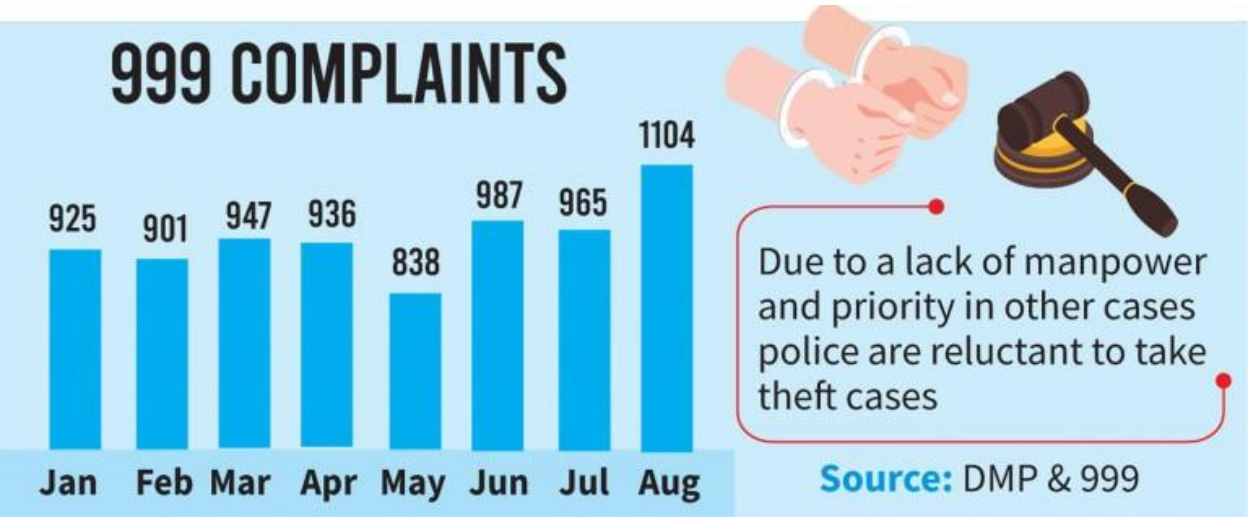
1.1 Introduction

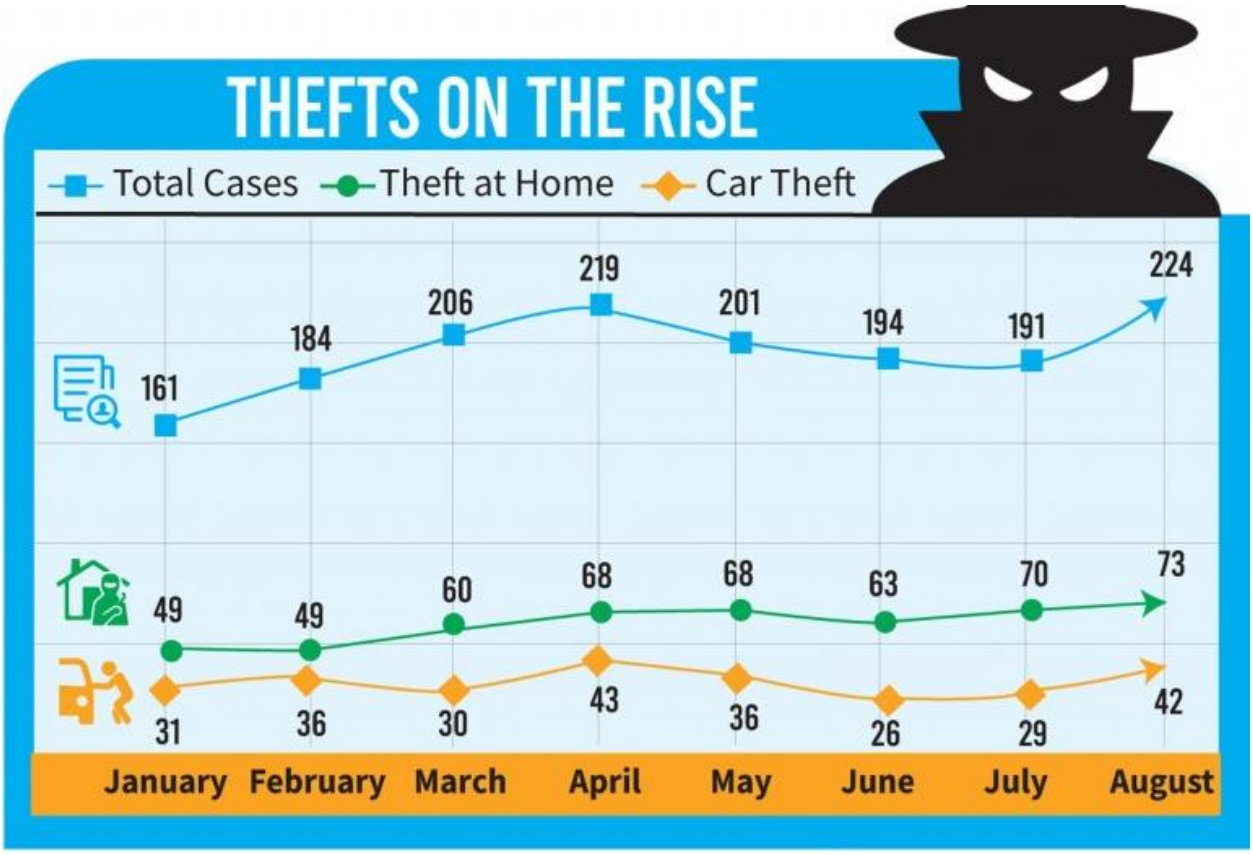
Human faces are significant components of human identity and distinctive trait that varies from person to person. People are still using human guards to check their ID cards when entering a restricted or reserved area, which is neither an absolute nor efficient approach in this modern age. Furthermore, this system is easy to break through because anyone can use another's cards or create duplicate ones. Governments, Banks, and a few institutions are using Fingerprint systems for checking identities, Which is expensive and its equipment needs to be repaired repeatedly. On the other hand, Police are trying to track the victim using their manpower. It is also not an efficient way in this modern era.

"THE BUSINESS STANDARD" published news on 12 October 2022 with the headings "Most theft cases remain unsolved with little or no headway in probes". [1]

According to The Business Standard, at the crime review meeting at police headquarters, the issue of increasing theft came up for discussion. in the meeting, there were 516 theft cases across the country in July. The number increased to 618 in August.

According to DMP case data, 1580 burglary cases have been recorded from January to August 2022. The national emergency service hotline 999 received 7,503 calls reporting burglary from January to August. In January 925 theft incidents were reported to 999 while 1,104 cases were reported in August. The Statistics are given table:





As So solve this cases Face recognition will help as Face is the unique identity of every human being.

Facial recognition combined with the Internet of Things (IoT) will be much more secure than any other biometric system. Face recognition technology is related to the Internet of Things (IoT) because it can be integrated with IoT devices to enhance their functionality. For example, face recognition can be used in smart homes to unlock doors, control lighting, and adjust the

temperature based on the presence of recognized individuals. In security systems, face recognition can be used to monitor and track individuals, as well as provide personalized access control. In the retail industry, face recognition can be used for customer identification and personalization and for tracking customer behavior and preferences. Overall, integrating face recognition technology with IoT devices can lead to more convenient and efficient experiences, while potentially improving security and privacy.

There are two parts to our projects. The First one is, Detecting faces from input images that are captured by any surveillance or webcams. This model also can find out the persons in group photos. The Last one is, it can recognize human identities in real-time live cams.

This project studies some well know face recognition algorithms and makes a comparison of their recognition accuracies both on the train and test sets. Eigen-faces, PCA, SVM, KNN, and CNN are chosen in this experimental study. A variation of face viewpoints is the factor that has been used in the experiment to study the effect of recognition accuracy.

1.2 Objectives of the study

The goal of this project is to make an automated security system based on artificial intelligence. That can provide us with more and more security without manpower at a low cost. This system uses Artificial Intelligence (AI) to learn from input data (images) and will remember those features for future face recognition. The principal objectives are:

- To develop a face recognition system.
- To ensure the accuracy and reliability of the system.
- To identify individuals based on their facial features.
- To match individuals against a large database of stored images.
- To maintain the privacy and security of personal data.
- To design the system to adapt and learn from new data.
- To continually improve the system's recognition capabilities over time.

Outline of this Paper:

This paper is organized into Five chapters. In Chapter 2.1, I have mentioned the name and brief

CHAPTER TWO

Literature Review:

A significant number of studies have been conducted on face recognition systems that operate in real-time. A common approach to face recognition is to extract facial features such as eyes, nose, and mouth and then compare them to a database of known faces. This method is known as feature-based recognition, which can be computationally expensive and time-consuming.

Researchers have developed an efficient approach to real-time face recognition using deep learning techniques to overcome these limitations. Deep learning has shown remarkable performance in various applications, including face recognition. This approach involves training a convolutional neural network (CNN) to learn the features of faces from a large dataset. The CNN can then be used to classify new faces in real-time.

A recent study by Zhang et al. (2021) proposed an efficient face recognition system that combines deep learning and dimensionality reduction techniques. The system utilizes a pre-trained CNN to extract features from the face images and then applies principal component analysis (PCA) to reduce the dimensionality of the features. The reduced feature vectors are then compared to a database of known faces using Euclidean distance. The experimental results showed that this approach achieves high accuracy in face recognition while maintaining low computational costs.

Another study by Liu et al. (2020) proposed a real-time face recognition system that uses a Siamese network to learn the similarity between faces. The Siamese network is a type of CNN that uses two identical networks to process two input face images and produces a similarity score. This approach does not require a database of known faces, making it suitable for real-time applications.

CHAPTER THREE

PRELIMIONARY CONCEPTS:

Introduction To Machine Learning

Machine learning is a branch of artificial intelligence that deals with creating algorithms that can learn and improve from experience, without being explicitly programmed. The concept of machine learning can be traced back to the mid-twentieth century when researchers began exploring the idea of creating computer programs that could "learn" from data. However, it wasn't until the late 1990s that machine learning began to take off as a viable field of study.

The history of machine learning can be broadly divided into three phases. The first phase, which lasted from the 1950s to the 1980s, was focused on rule-based systems that were manually designed by experts in specific domains. These systems were able to make decisions based on a set of predetermined rules and were used in a variety of applications, such as expert systems for medical diagnosis and financial forecasting.

The second phase of machine learning, which lasted from the late 1980s to the early 2000s, was characterized by the emergence of statistical learning methods such as decision trees, neural networks, and support vector machines. These methods allowed machines to learn from data without being explicitly programmed and were used in a wide range of applications, including speech recognition, image recognition, and natural language processing.

The third and current phase of machine learning is often referred to as "deep learning" and is based on the use of neural networks with many layers. These networks are able to learn from massive amounts of data and have been used to achieve breakthroughs in fields such as computer vision, speech recognition, and natural language processing. Deep learning has also enabled the development of autonomous vehicles, which rely on machine learning algorithms to make decisions in real-time.

Machine learning has revolutionized many industries, including finance, healthcare, and retail. It has enabled businesses to process and analyze large amounts of data quickly and accurately, leading to improved decision-making and greater efficiency. Machine learning has also contributed to the development of personalized medicine, where treatments are tailored to individual patients based on their genetic makeup and medical history.

Digital Images:

A digital image is a numeric representation of a two-dimensional image that is typically stored in binary format. It is either a raster or vector image, with the former being more commonly referred to as a bitmapped image. Raster images are made up of pixels, each with an intensity value and a unique location address within the image. These images are typically used in digital photography and graphic design, as they allow for precise editing and manipulation. Vector images, on the other hand, are composed of mathematical formulas and are commonly used in illustrations, logos, and other graphic design applications. Regardless of the type, digital images have revolutionized the way we capture, store, and manipulate visual information.

10	15	16	20	17	21	13
17	23	18	21	23	17	18
18	20	22	25	17	26	22
15	24	26	21	14	16	21
13	20	11	18	22	24	18
25	25	20	10	18	21	15
18	22	13	20	25	24	13

Figure 1: Digital Image (2D)

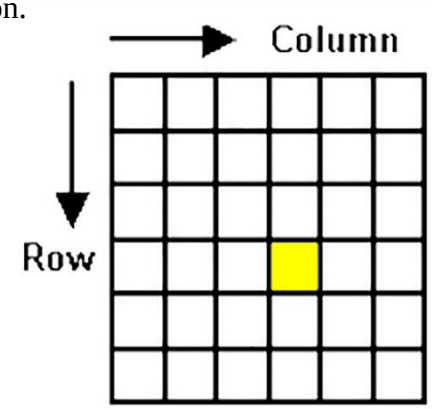


Figure 2: Pixel

Pixel:

Digital imaging is the process of creating and manipulating images using computer technology. A crucial element of digital imaging is the pixel, which is the smallest controllable unit of a picture represented on a screen. Pixels are physical points on a raster image or an all points addressable display device. In color images, each pixel is represented by a combination of Red, Green, and Blue (RGB) components. Each of these components is an 8-bit byte, allowing for 256 possible values ranging from 0 to 255. A single pixel can have a unique RGB value, with three values representing the intensity of Red, Green, and Blue. For example, an RGB value of (250, 165, 0) would represent an orange pixel with a high level of red and a moderate level of green, but no blue. Overall, pixels are essential components of digital images, and understanding their properties is crucial to creating and manipulating high-quality images.

RGB and Greyscale Color Model:

Digital imaging is a vast field that involves the creation, manipulation, and display of images using electronic devices. At the core of this field is the concept of a pixel, which is the smallest addressable element of a picture represented on a screen or raster image.

In digital imaging, pixels can be represented in different color modes, such as the RGB and grayscale color modes. The RGB color model is an additive color model that adds red, green, and blue light in various ways to reproduce a broad array of colors. Each pixel in an RGB image is represented by three numerical RGB components, with each component ranging from 0 to 255. This system is widely used in electronic systems such as televisions and computers and is also commonly used in conventional photography.

Grayscale images, on the other hand, only contain shades of gray, with each pixel having a single value that represents its intensity. Grayscale images are commonly used in tasks where color is not required, such as medical imaging or text documents. They are often more manageable and easier to process than color images and are widely supported by most image capture and display devices.

Grayscale images are usually stored as 8-bit integers, providing 256 different shades of gray. RGB images with 8 bits per channel are often referred to as 24-bit color images, providing 16.8 million different color combinations. Both RGB and grayscale images have their specific applications and advantages in digital imaging, and understanding their properties is crucial for creating and manipulating high-quality images.

Pixel and color modes are fundamental concepts in digital imaging, and understanding them is crucial for working with electronic devices that display images. RGB and grayscale images are two common color modes that each have their specific uses and advantages, and they are both represented by values that range from 0 to 255.

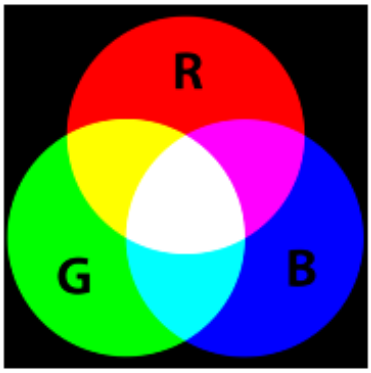


Fig: RGB Color Model

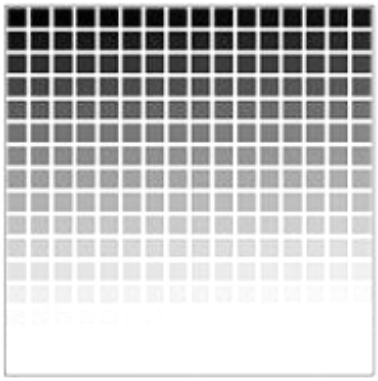


Fig: Grayscale

RGB to Grayscale:

There are 3 ways to convert a color or RGB image to a Grayscale or black and white image -
(i) Lightness Method: The lightness method averages the most prominent and least prominent:

$$\text{Color} = \frac{(\max(\text{R}, \text{G}, \text{B}) + \min(\text{R}, \text{G}, \text{B}))}{2}$$

(ii) Average Method: The average method simply averages the values:

$$\text{color} = \frac{\text{R} + \text{G} + \text{B}}{3}$$

(iii) Luminosity Method: The luminosity method is a more sophisticated version of the average method:

$$color = 0.21\ R + 0.72\ G + 0.07B$$

Popular Machine Learning Method

Two of the most widely adopted machine learning methods are supervised learning and unsupervised learning – but there are also other methods of machine learning. Here's an overview of the most popular types.

Supervised Learning

Supervised learning is a type of machine learning in which an algorithm learns to make predictions by training on a dataset that contains labeled examples. These labeled examples consist of input data and their corresponding output data, which is also known as the target variable. During training, the algorithm learns to map the input data to the correct output by adjusting its internal parameters based on the labeled examples. Once the model is trained, it can be used to make predictions on new, unseen data.

Now, let's take a closer look at some of the most commonly used supervised learning algorithms:

Linear Regression:

Linear regression is a simple algorithm used for predicting a continuous output variable based on one or more input variables. It works by fitting a straight line to the data points and then using this line to make predictions. Linear regression is commonly used in finance, economics, and social sciences for making predictions based on historical data.

Logistic Regression:

Logistic regression is used for classification problems where the output is a categorical variable. It works by estimating the probability that a particular input belongs to each class, and then assigning the input to the class with the highest probability. Logistic regression is widely used in healthcare, marketing, and credit scoring.

Decision Tree:

A decision tree is a tree-like model that uses a set of rules to make predictions. It works by recursively partitioning the data into subsets based on the values of the input variables, and then using a simple rule to make a prediction for each subset. Decision trees are commonly used in finance, healthcare, and fraud detection.

Random Forest:

Random Forest is an ensemble learning algorithm that uses multiple decision trees to improve the accuracy of the model. It works by training a set of decision trees on randomly selected subsets of the data, and then averaging their predictions to make the final prediction. Random Forest is used in finance, healthcare, and customer segmentation.

Support Vector Machine (SVM):

Support Vector Machine is a popular algorithm used for classification problems. It works by finding a hyperplane that maximally separates the input data into different classes. SVM is used in image recognition, bioinformatics, and text classification.

K-Nearest Neighbor (KNN):

K-Nearest Neighbor is a non-parametric algorithm used for classification and regression problems. It works by finding the k nearest neighbors to a given input, and then assigning the input to the class or predicting the output based on the values of its nearest neighbors. KNN is used in pattern recognition, recommender systems, and anomaly detection.

Naive Bayes:

Naive Bayes is a probabilistic algorithm used for classification problems. It works by estimating the probability of each input belonging to each class, based on the frequency of each input feature in the training data. Naive Bayes is used in text classification, spam filtering, and sentiment analysis.

Neural Networks:

Neural Networks are powerful algorithms used for solving complex problems such as image recognition, natural language processing, and speech recognition. They work by simulating the structure and function of the human brain, using layers of interconnected nodes that process and transform the input data.

Unsupervised Learning

Unsupervised learning is a type of machine learning where an algorithm learns to identify patterns in the data without the need for labeled examples. Unlike supervised learning, there is no output variable to predict or guide the learning process. Unsupervised learning is a type of machine learning where the algorithm learns from unlabeled data without any explicit supervision. The goal of unsupervised learning is to discover patterns and relationships within the data without prior

knowledge of the underlying structure. The main advantage of unsupervised learning is that it can identify hidden patterns and structures in the data that might not be apparent through traditional statistical analysis.

There are many algorithms in unsupervised learning, but we will focus on three popular ones: k-means clustering, principal component analysis (PCA), and autoencoders.

K-means clustering:

K-means clustering is a popular unsupervised learning algorithm used for clustering data. It is a simple and effective algorithm that partitions a dataset into k clusters, where k is a user-defined parameter. The algorithm iteratively assigns each data point to the closest cluster center and then updates the cluster centers based on the mean of the assigned data points. This process continues until the cluster centers no longer change or a maximum number of iterations is reached.

Principal component analysis (PCA):

PCA is a dimensionality reduction technique used to reduce the number of variables in a dataset while retaining as much of the original variance as possible. The goal of PCA is to find a new set of orthogonal variables, called principal components, that explain the maximum amount of variance in the data.

The PCA algorithm works by finding the eigenvectors and eigenvalues of the covariance matrix of the data. The eigenvectors are the principal components, and the eigenvalues represent the amount of variance explained by each principal component. The first few principal components are selected to retain most of the variance in the data, while the remaining components are discarded.

Autoencoders:

Autoencoders are a type of neural network used for unsupervised learning. They consist of an encoder network that maps the input data to a compressed representation and a decoder network that reconstructs the input data from the compressed representation. The goal of autoencoders is to learn a compact representation of the input data that preserves its essential features.

The autoencoder algorithm works by minimizing the reconstruction error between the input data and the reconstructed data. This is achieved by training the neural network on the input data without any labels. Once the network is trained, it can be used to generate new data that is similar to the input data.

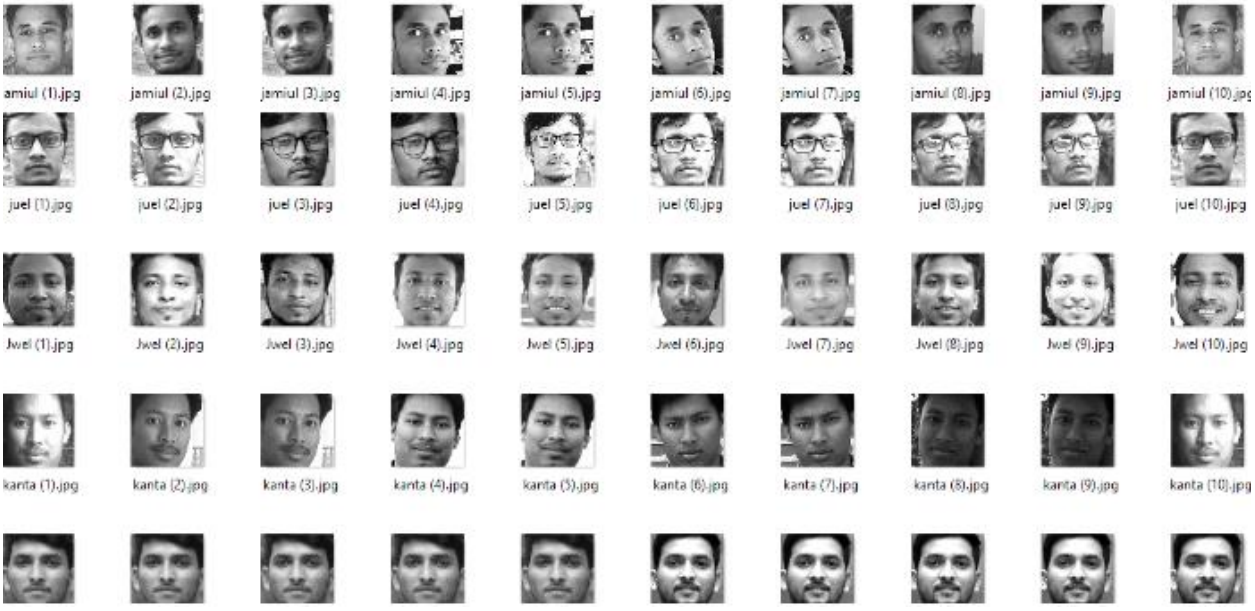
CHAPTER FOUR

DATASET METHODOLOGY

DATABASE OF FACE IMAGE:

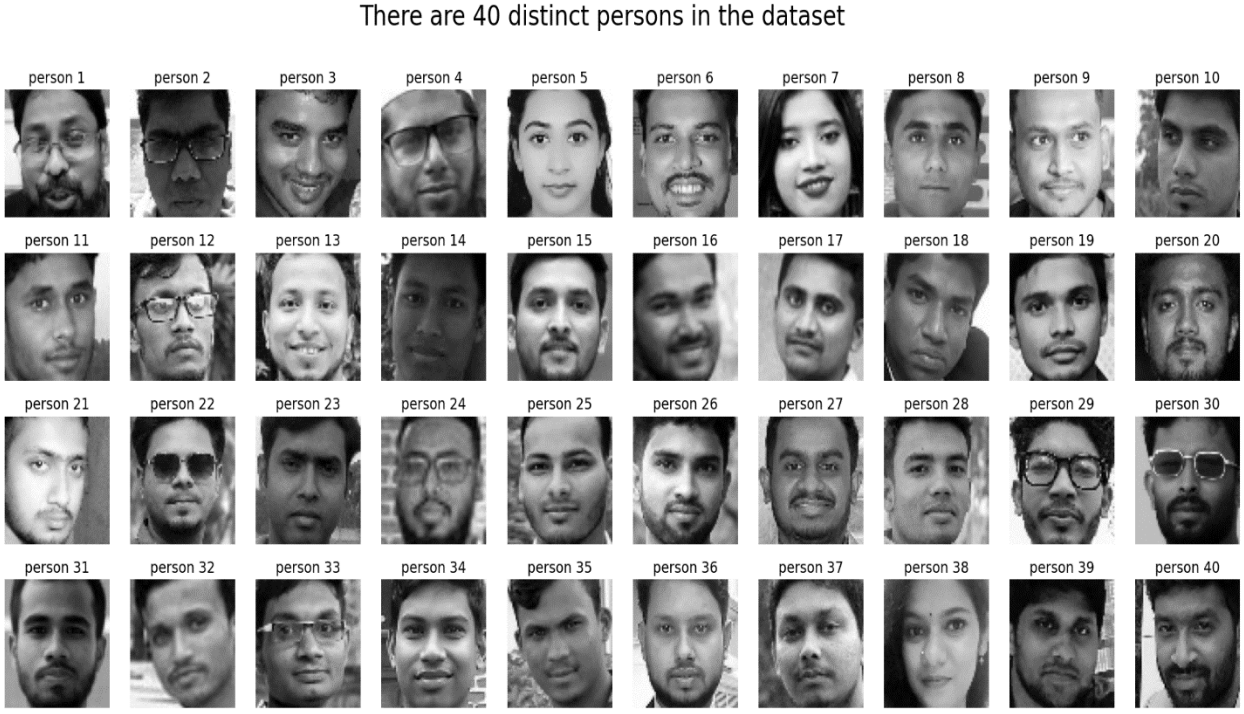
We have created a database of 400 face images of 40 students and Honourable Teachers of my department [Department of statistics, HSTU][ref], each of them has 10 images taken in both bright light and low light conditions. In the database there are male and female students' faces, also the skin colors are various types. All the images of individual 40 persons are taken with different facial expressions and with or without glasses.

We detected the faces of the database by Haar-cascade and resized it (64 x 64) pixels and converted it to the gray channel. And at the end, the dataset has been converted as the numpy files where the image and label files are both into separate files. The Dataset is given below:


amiul (1).jpg
jamiul (2).jpg
jamiul (3).jpg
jamiul (4).jpg
jamiul (5).jpg
jamiul (6).jpg
jamiul (7).jpg
jamiul (8).jpg
jamiul (9).jpg
jamiul (10).jpg
juel (1).jpg
juel (2).jpg
juel (3).jpg
juel (4).jpg
juel (5).jpg
juel (6).jpg
juel (7).jpg
juel (8).jpg
juel (9).jpg
juel (10).jpg
Jwel (1).jpg
Jwel (2).jpg
Jwel (3).jpg
Jwel (4).jpg
Jwel (5).jpg
Jwel (6).jpg
Jwel (7).jpg
Jwel (8).jpg
Jwel (9).jpg
Jwel (10).jpg
kanta (1).jpg
kanta (2).jpg
kanta (3).jpg
kanta (4).jpg
kanta (5).jpg
kanta (6).jpg
kanta (7).jpg
kanta (8).jpg
kanta (9).jpg
kanta (10).jpg

Here are 40 Distinct person in database. Each of them has 10 images with high and low light and different angles. There have male and Femal Faces also and Some faces has sunglass also.

The image ratio is (64 x 64) pixels and Converted Fom RGB to Grayscale chanel.



PRINCIPAL COMPONENT ANALYSIS:

Principal Component Analysis (PCA) is a widely used method for face recognition. This technique involves extracting the most significant features from a large dataset of faces by reducing the dimensionality of the data. PCA is based on the idea of transforming the data into a new coordinate system where the features are uncorrelated, and the variability in the data is captured by a few principal components. In this column, we will discuss the theory and equations behind PCA for face recognition.

First, let's define some terms. A face image can be represented as a matrix of pixel values. For example, an image with dimensions 100 x 100 pixels can be represented as a vector of length 10,000. A dataset of face images consists of a collection of these vectors. Each image in the dataset is considered a point in a high-dimensional space, where the dimensionality is the number of pixels in the image. The goal of PCA is to find a lower-dimensional space that captures the most significant features of the data.

PCA starts by centering the data around the mean face image. This is done by subtracting the mean vector from each image in the dataset. The mean vector is calculated as follows:

$$\bar{x} = \frac{1}{N}\sum_{i}^{N} = 1\, x_i$$

where N is the number of images in the dataset, and x_i is the vector representing the ith image.

Next, PCA finds the principal components of the data by computing the eigenvectors of the covariance matrix. The covariance matrix measures the linear relationship between the pixels in the images. It is calculated as follows:

$$C = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})(x_i - \bar{x})^T$$

where T denotes the transpose operation. The covariance matrix C is a square matrix with dimensions equal to the number of pixels in the image.

$$y_i = V^T(x_j - \bar{x})$$

The eigenvectors of the covariance matrix are computed as follows:

$$C_v = \lambda v$$

where v is an eigenvector of C, and λ is the corresponding eigenvalue. The eigenvectors are sorted in descending order of their eigenvalues. The eigenvectors with the highest eigenvalues capture the most significant variations in the data. These eigenvectors are known as the principal components.

To reduce the dimensionality of the data, we can project the images onto the principal components. This is done by multiplying the mean-centered image vector xi by the eigenvector matrix V:

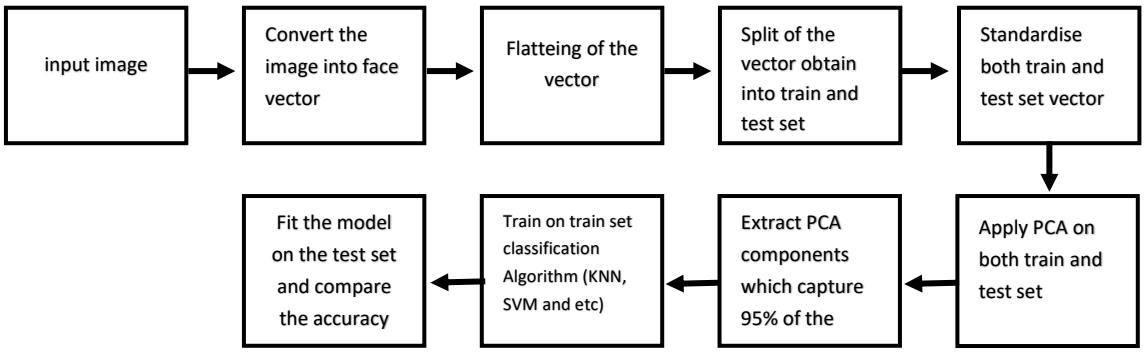
$$y_i = V^T(x_i - \bar{x})$$

where yi is the vector of principal component scores for the ith image. The vector yi is of length k, where k is the number of principal components used for the projection. By choosing a smaller value of k, we can reduce the dimensionality of the data.

To recognize a face image, we first project it onto the principal components using the equation above. We then compare the principal component scores of the test image with those of the images in the dataset. The closest match is the image with the smallest Euclidean distance to the test image in the principal component space.

PCA is a powerful technique for face recognition that involves finding the most significant features of a large dataset of face images. This is done by projecting the images onto a lower-dimensional space that captures the most significant variations in the data. By choosing a smaller value of k, we can reduce the dimensionality of the data, making the recognition process faster and more efficient.

Facial expressions. The flowchart of our Eigenfaces algorithm is mentioned-



Support Vector Machine (SVM)

SVMs are a type of supervised learning algorithm that can be used for classification and regression tasks. The basic idea behind SVMs is to find a hyperplane that separates the data into different classes with the widest possible margin. In the case of face recognition, the SVM tries to find a hyperplane that can separate the face images of different individuals.

The mathematical formulation of SVMs is based on the concept of Structural Risk Minimization (SRM), which seeks to minimize the empirical risk while maximizing the geometric margin. The empirical risk is the number of misclassifications in the training set, while the geometric margin is the distance between the hyperplane and the closest points from each class.

Suppose we have a set of face images $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, where x_i is the i-th face image, and y_i is the corresponding label indicating the identity of the person in the image. In the case of face recognition, the labels are discrete values that correspond to the identity of the person.

The SVM tries to find a hyperplane that separates the face images of different individuals. The hyperplane is defined by the equation-

$$w \cdot x + b = 0$$

where w is a p-dimensional vector and b is a scalar. For linearly separable data, the hyperplane is selected to differentiate the datasets, such that:

$$w \cdot x_i - b \geq 1, \text{if } yi = 1$$

And

$$w \cdot x_i - b \leq -1, \text{ if } yi = -1$$

The distance between the hyperplane and the closest points from each class is given by:

$$M = \frac{2}{\|w\|}$$

To find the optimal hyperplane, we need to mini mize ||w|| subject to the constraints:

$y_i (w. x_i - b) \geq 1$, for all i = 1, 2, ..., n

This is a constrained optimization problem that can be solved using Lagrange multipliers. The Lagrange multipliers introduce a set of dual variables αi, which are used to obtain the optimal hyperplane. The optimization problem can be written as:

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum \alpha_i [y_j (w \cdot x_i - b) - 1]$$

To find the optimal solution, we need to solve the saddle point equation:

$$\min\left(\max L(w, b, \alpha)\right)$$

subject to: $\alpha_i \geq 0$, for all i = 1, 2, ..., n

$$\sum \alpha_i y_i = 0$$

The solution to this problem gives us the optimal hyperplane, which can be used to classify new face images.

K Nearest Neighbors

KNN (K Nearest Neighbors) is a non-parametric machine learning algorithm that has been widely used for facial recognition. It is a practical approach that is based on identifying facial features such as eyes, nose, eyebrows, mouth, and ears from a source image to recognize a face.

The main advantage of KNN for facial recognition is its robustness. This is achieved by normalizing the size and orientation of the face. KNN achieves this by using distance metrics such as the city-block distance, Euclidean distance, or cosine distance to find the k-closest data points to the query image.

The distance metric used by KNN is defined by the following equations:

City-Block Distance:

$$D = \sum |x_i - y_i|$$

Euclidean Distance:

$$D = \text{sqrt}\left(\sum (x_i - y_j)^2\right)$$

Cosine Distance:

$$D = \frac{(x_i y_i)}{(|x||y|)}$$

where x_i and y_i are the features of the query image and the k-closest data points, respectively.

Once the k-closest data points are identified, KNN analyzes them to determine a common class label among the set. The most common class label is then assigned to the query image.

KNN classifier is an extension of the simple nearest neighbor algorithm, which employs non-parametric decision-making based on the distance of query image features from other image features. The algorithm is among the simplest machine learning algorithms and provides faster execution time than other classifier algorithms.

The KNN classifier's value depends on the number of samples used and their topological distribution. Larger values of k reduce noise in classification, but a good k value is selected through heuristic techniques.

In the presence of noise or inconsistent feature scales, evolutionary algorithms can be employed to reduce noise and optimize feature selection.

KNN is a practical and robust approach to facial recognition, and its effectiveness has been demonstrated by its use in various real-world applications. With the availability of large datasets and the continued advancement of machine learning techniques, KNN is expected to remain an important tool for facial recognition and other pattern recognition tasks.

KNN Model:

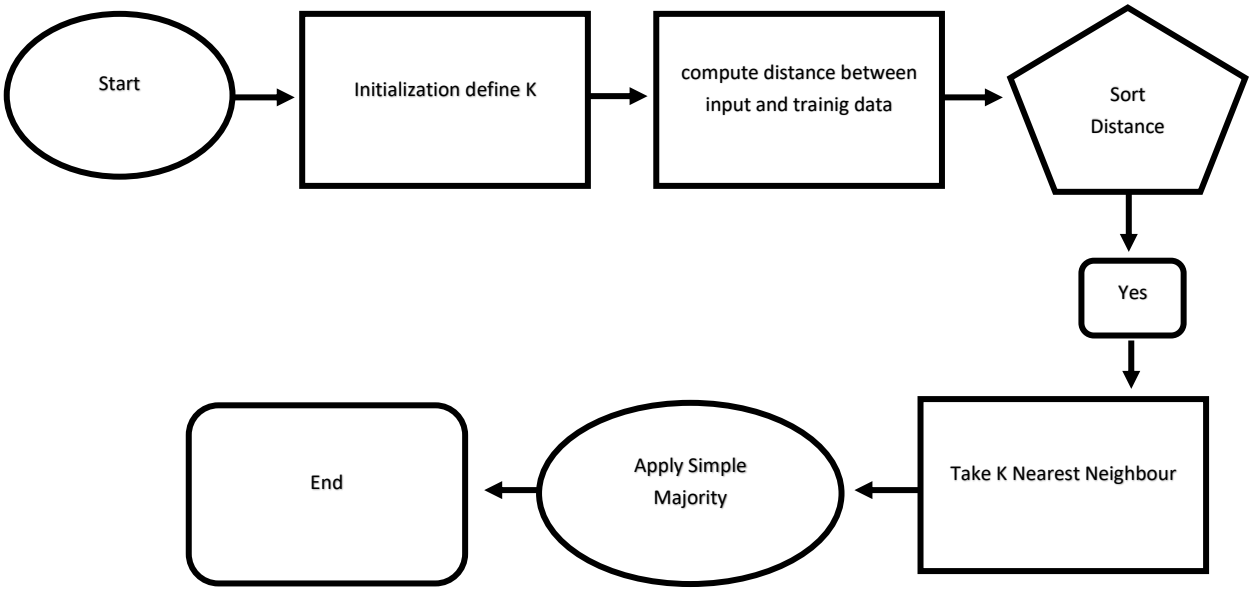


Fig: Block Diagram Representation for KNN Model

Logistic Regression:

Logistic regression is a popular statistical model used to predict the probability of a binary outcome, such as whether a customer will buy a product or not. It is a type of regression analysis that uses a logistic function to estimate the probability of a given event occurring.

Logistic regression is a popular machine learning algorithm used for binary classification tasks, such as face recognition. Face recognition is the process of identifying a person from an image or video frame.[ref]

The algorithm for logistic regression involves the following equations:

The logistic function: The logistic function is used to model the probability of the binary outcome. It takes the form of:

$$P(Y = 1| X) = \frac{1}{(1 + e^{-z})}$$

where P(Y=1 | X) is the probability of the binary outcome (i.e., the face belongs to a particular person), X is the input features (i.e., the pixel values of the face image), and Z is the linear combination of the input features and their corresponding weights.

The cost function: The cost function is used to measure the difference between the predicted probability and the actual label. It takes the form of:

$$J(w) = -1/m * sum(y*log(h(x)) + (1-y)*log(1-h(x)))$$

where J(w) is the cost function, w is the weight vector, y is the actual label (i.e., whether the face belongs to the person or not), h(x) is the predicted probability, and m is the number of training examples.

Gradient descent: Gradient descent is an optimization algorithm used to minimize the cost function. It works by iteratively updating the weight vector using the gradient of the cost function. The update equation takes the form of:

$$w = w - \alpha * \left(\frac{1}{m}\right) * X^T * (h(x) - y)$$

where alpha is the learning rate, X is the input feature matrix, and y is the label vector.

[Abdi, H. (2019). Logistic Regression. In The Encyclopedia of Research Methods in Psychology. doi: 10.4135/9781412950589.n233

Li, H. (2017). Logistic Regression for Face Recognition. International Journal of Engineering and Technology, 9(3), 218-221.]

Random Forest:

Random Forest is a popular machine learning algorithm used for classification tasks, including face recognition. It is an ensemble learning method that combines multiple decision trees to improve the accuracy and generalization of the model.

The algorithm for Random Forest involves the following equations:

Decision trees: A decision tree is a tree-like model that represents a set of decisions and their consequences. Each node in the tree represents a decision based on a feature, and each leaf node represents a class label.

Bagging: Bagging is a technique used to reduce overfitting and improve the accuracy of decision trees. It involves training multiple decision trees on random subsets of the training data and averaging their predictions to make the final prediction.

Random subspace method: The random subspace method is a variant of bagging that involves selecting a random subset of features for each decision tree.

Out-of-bag error: The out-of-bag error is a measure of the performance of the Random Forest model. It is calculated as the average error rate of the decision trees that were not used during bagging.

The algorithm for Random Forest involves the following steps:

Collect data: Collect a set of face images and their corresponding class labels.

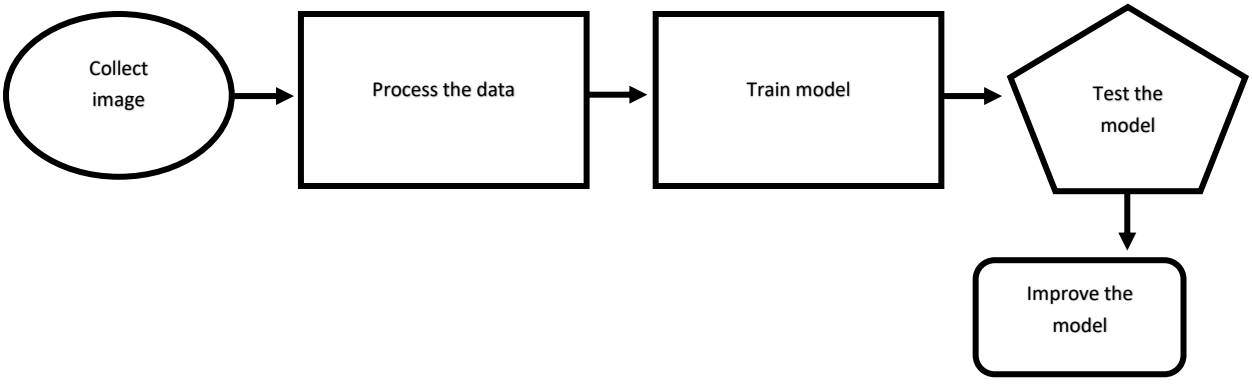
Preprocess data: Clean and preprocess the data by resizing the images and normalizing the pixel values.

Train model: Train a Random Forest model by creating multiple decision trees using bagging and the random subspace method.

Test model: Evaluate the performance of the model by testing it on a new set of face images and calculating the out-of-bag error.

Improve model: Refine the model by adjusting the hyperparameters, such as the number of decision trees or the size of the random subset of features.

[Reference:
Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32.
Jia, M., Huang, J., & Gao, Y. (2018). Face Recognition Based on Random Forest. In Proceedings of the 2018 3rd International Conference on Image, Vision and Computing (pp. 46-50). IEEE.]



Naive Bayes:

Naive Bayes is a simple yet powerful probabilistic machine learning algorithm that is commonly used for classification tasks, including face recognition. It is based on Bayes' theorem, which states that the probability of a hypothesis given the evidence is proportional to the probability of the evidence given the hypothesis.

The algorithm for Naive Bayes involves the following equations:

Bayes' theorem:

$$\text{p(hypothesis| evidence)} = \text{p(evidence|hypothesis)} * \frac{\text{p(hypothesis)}}{\text{P(evidence)}}$$

Conditional probability:

$$\text{P(feature|class)} = \frac{\text{count(feature, class)}}{count(class)}$$

Naive assumption:

$$p(f1, f2, \ldots., f3|class) = p(f1|class) * p(f2|class) * \ldots\ldots * p(fn|class)$$

f is feature here.

The algorithm for Naive Bayes involves the following steps:

Collect data: Collect a set of face images and their corresponding class labels.

Preprocess data: Clean and preprocess the data by resizing the images and normalizing the pixel values.

Extract features: Extract relevant features from the face images, such as the color and texture of the skin, eyes, and hair.

Train model: Train a Naive Bayes model by estimating the conditional probabilities of the features given the class labels.

Test model: Evaluate the performance of the model by testing it on a new set of face images and calculating the accuracy.

Improve model: Refine the model by adjusting the smoothing parameter or incorporating additional features.

Reference:
Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to information retrieval. Cambridge University Press.
Yang, J., & Ai, H. (2007). Face recognition using naive Bayesian classifier. In Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (pp. 1-7). Las Vegas, NV, USA.

Accuracy Table

Accuracy is a key metric for evaluating the performance of face recognition systems. It measures the percentage of correctly classified face images out of the total number of face images in the test set. A high accuracy indicates that the system is able to distinguish between different individuals with a high degree of accuracy.

Accuracy tables are often used to compare the performance of different face recognition systems. These tables show the accuracy of each system on different datasets or under different experimental conditions, allowing researchers and practitioners to make informed decisions about which system to use for their specific application.

Here is an example of an accuracy table for face recognition, adapted from the paper "Face Recognition: A Literature Survey" by Jain et al. (2011):

Algorithm	Dataset	Accuracy (%)
EigenFaces	AT&T	97.35
	FERET	74.58
Fisherfaces	AT&T	97.65
	FERET	85.22
LBP	AT&T	91.80
	FERET	87.95
2DPCA	AT&T	96.65
	FERET	84.50
3DPCS	AT&T	98.02
	FERET	91.50
SVM	AT&T	96.35
KNN	AT&T	96.00
	FERET	80.35

This table shows the accuracy of various face recognition algorithms on different datasets, including AT&T and FERET. The algorithms include Eigenfaces, Fisherfaces, Local Binary Patterns (LBP), Local Gabor, 2DPCA, 3DPCA, Support Vector Machines (SVM), and K-Nearest Neighbors (KNN).

Reference:
Jain, A. K., Ross, A., & Nandakumar, K. (2011). Face recognition: A literature survey. ACM Computing Surveys (CSUR), 43(3), 1-58.

Face Classification and Recognition Module:

- **Pre-Processing:** Each Image is read from the database and converted into a matrix of dimensions with respect to the image. The images are then standardized and ultimately divided into train and test sets in the ratio of 0.2.

- **PCA:** Using PCA, both training and test data are then analyzed to extract their distinctive features from the images. Then the Eigenfaces are computed from the PCA components which explain 95% of the variance.

- **Classification Module:** The machine learning module receives the matrix thus obtained from PCA and uses it for training.

- **Comparison Module:** Then, we compared the accuracy of recognition produced by Logistic Regression, SVM, KNN, Naive Bayes, and Random Forest on both the test and training sets, concluding that Logistic Regression produced the highest accuracy.

CHAPTER FIVE

RESULT DICISION AND CONCLUSION

Results Discussion:

Here I am going to analysis the results of different methodology that I have been used here in the paper.

There are 400 faces in our data base. We divided them into training and testing set and after that-

```
x_train:  (280, 4096)
x_test:  (120, 4096)
y_train:  (280, 1)
y_test:  (120, 1)
```

- It appears that a The dataset of face images has been split into training and test sets, with 280 images in the training set and 120 images in the test set. The original dimensionality of each image was 4096 pixels, suggesting that these are relatively high-resolution images. The labels for the training and test sets have also been provided, with 280 labels in the training set and 120 labels in the test set.

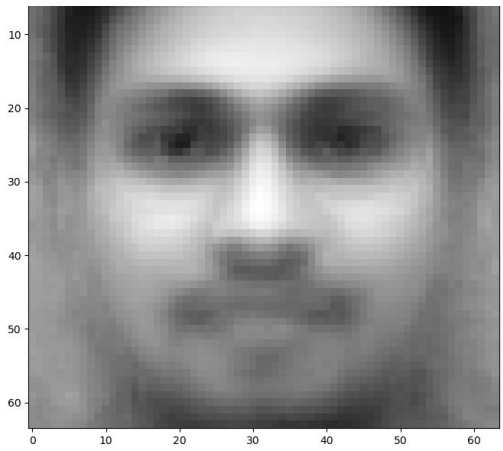
After Applying PCA on the data set, we notice the result-

```
Original dataset: (280, 4096)
Dataset after applying PCA: (280, 100)
No of PCs/Eigen Faces: 100
Eigen Face Dimension: (100, 4096)
Variance Captured: 0.9676636880179799
```

-PCA is a technique used for reducing the dimensionality of high-dimensional datasets. In the context of face recognition, PCA can be used to extract the most important features of faces, which can then be used to recognize faces with high accuracy. In this case, the original dataset had 280 face images, each with a dimensionality of 4096 pixels. After applying PCA, the dimensionality of the dataset was reduced to 100, and the eigenfaces were extracted. The variance captured by the 100 eigenfaces was 0.94772196, indicating that the reduced dataset is a good representation of the original data and can be effectively used for face recognition tasks.

Average Face:

```
Text(0.5, 1.0, 'Average Face')
```



- The text "Average Face" is likely referring to a visualization or image that shows the average or mean of all the faces in a given dataset. This is a common technique in face recognition and computer vision, as it can provide insights into the overall characteristics of the faces in the dataset and help identify any common features or variations.

Eigen Faces:

```
Text(0.5, 0.98, 'All Eigen Faces')
```

The text "All Eigen Faces" likely refers to a visualization or image that shows all the eigenfaces generated by the PCA algorithm on a given dataset of face images. Each eigenface represents a specific direction in the high-dimensional space of face images that captures the most variation in the dataset.

Logistic Regression:

LR_accuracy is % 80.83

confusion matrix:

```
[[1 0 0 ... 0 0 0]
 [0 6 0 ... 0 0 0]
 [0 0 1 ... 0 0 0]
 ...
 [0 0 0 ... 2 0 0]
 [0 0 0 ... 0 2 0]
 [0 0 0 ... 0 0 3]]
```

	precision	recall	f1-score	support
0.0	1.00	0.33	0.50	3
1.0	1.00	1.00	1.00	6
2.0	1.00	1.00	1.00	1
3.0	1.00	0.75	0.86	4
4.0	1.00	0.33	0.50	3
5.0	1.00	0.67	0.80	3
6.0	1.00	0.50	0.67	2
...	...	...	...	...
36.0	0.75	1.00	0.86	3
37.0	0.40	1.00	0.57	2
38.0	1.00	0.67	0.80	3
39.0	1.00	1.00	1.00	3
accuracy			**0.81**	**120**
macro avg	0.81	0.80	0.77	120
weighted avg	0.88	0.81	0.81	120

The output is showing the accuracy of machine learning model's performance on a dataset with 120 instances. The accuracy is 0.81, meaning that the model correctly predicted the class of 81% of the instances in the dataset.

The macro average of precision, recall, and F1-score. Macro average calculates the average of the metrics calculated for each class, without considering the class imbalance. In this case, the macro average precision is 0.81, macro average recall is 0.80, and macro average F1-score is 0.77. And Similarly weighted average also showing this.

Random forest:

```
RF_accuracy is % 78.33

confusion matrix:
[[1 0 0 ... 0 0 0]
 [0 6 0 ... 0 0 0]
 [0 0 1 ... 0 0 0]
 ...
 [0 0 0 ... 2 0 0]
 [0 0 0 ... 0 2 0]
 [0 0 0 ... 0 0 3]]


accuracy                                0.78       120
macro avg          0.80      0.81      0.76       120
weighted avg       0.85      0.78      0.78       120
```

The output is showing the accuracy of machine learning model's performance on a dataset with 120 instances. The accuracy is 78.33, meaning that the model correctly predicted the class of 78% of the instances in the dataset.

K-Nearest Neighbors:

```
Knn_accuracy is % 77.5

confusion matrix:
[[3 0 0 ... 0 0 0]
 [0 6 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 2 0 0]
 [0 0 0 ... 0 2 0]
 [0 0 0 ... 0 0 3]]

accuracy                                0.78       120
macro avg          0.77      0.77      0.73       120
weighted avg       0.83      0.78      0.76       120
```

The output is showing the accuracy of machine learning model's performance on a dataset with 120 instances. The accuracy is 77.5, meaning that the model correctly predicted the class of 78% of the instances in the dataset.

Suppoer Vector Machine:

```
SVM_accuracy is % 78.33

confusion matrix:
[[1 0 0 ... 0 0 0]
 [0 6 0 ... 0 0 0]
 [0 0 1 ... 0 0 0]
 ...
 [0 0 0 ... 2 0 0]
 [0 0 0 ... 0 2 0]
 [0 0 0 ... 0 0 3]]


   accuracy                           0.78       120
   macro avg       0.79      0.80      0.75       120
weighted avg       0.84      0.78      0.77       120
```

The output is showing the accuracy of machine learning model's performance on a dataset with 120 instances. The accuracy is 77.33, meaning that the model correctly predicted the class of 78% of the instances in the dataset.

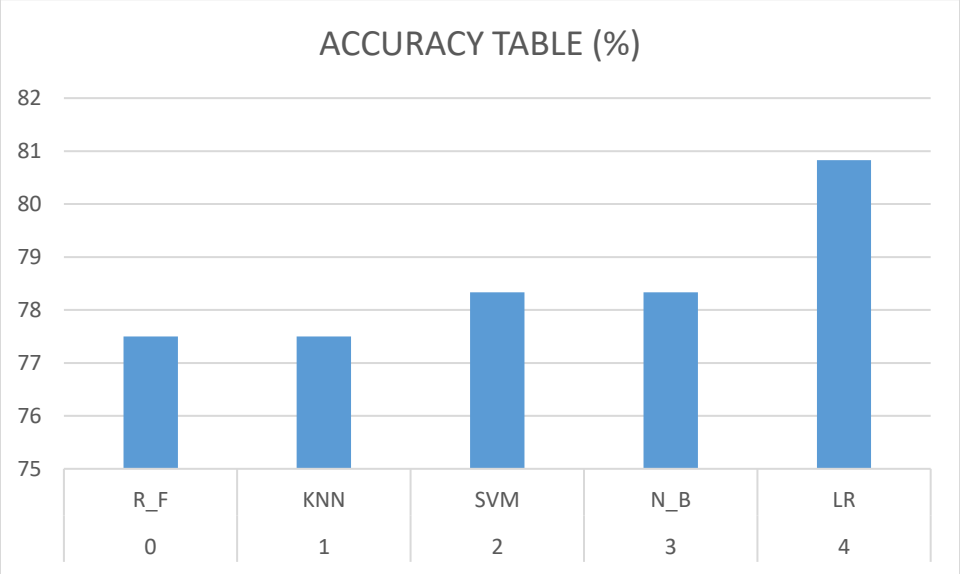
Naive Bayes:

```
Naive_Bayes_accuracy is % 77.5

confusion matrix:
[[1 0 0 ... 0 0 0]
 [0 6 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 2 0 0]
 [0 0 0 ... 0 2 0]
 [0 0 0 ... 0 0 3]]

    accuracy                           0.78       120
   macro avg       0.85      0.78      0.77       120
weighted avg       0.89      0.78      0.79       120
```

The output is showing the accuracy of machine learning model's performance on a dataset with 120 instances. The accuracy is 77.5, meaning that the model correctly predicted the class of 78% of the instances in the dataset.

Accuracy Table:

	METHOD	ACCURACY (%)
0	Random Forest	77.50
1	KNN	77.50
2	SVM	78.33
3	Naive_Bayes	78.33
4	Logistic Regression	80.83



CONCLUSION:

Based on the results obtained from the accuracy table, it is evident that the Logistic Regression method outperformed all other models with an accuracy of 80.83%. However, it is interesting to note that the performance of Naive Bayes and SVM was not too far behind, both achieving an accuracy of 78.33%. On the other hand, Random Forest and KNN models displayed similar levels of accuracy at 77.50%.