**Face recognition system to compare KNN performance**

Name: Aafaq Saleh Al Shidhani

**Introduction**

Face recognition system to compare KNN performance using different normalization techniques, for different values of K.

Face Recognition System is a computer application for automatically identifying or verifying a person from a digital image or a video frame. Face Recognition provides a more direct and friendly identification method and it is more acceptable to users as compared to other biometric methods. Because of variation in face pose angle, illumination, expression and occlusion there are many challenges in face recognition. Face appearance changes drastically with change in facial pose because of misalignment as well as hiding of many facial features and hence recognition of faces under pose variations is a difficult problem.

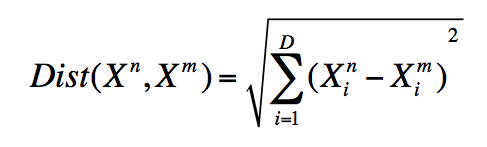
In this project we are trying to create a Facial Recognition System using K Nearest Neighbor algorithm, apply different normalization techniques and different values of K to see their effect on KNN. We will be using kernel\*, NumPy and some other python packages.

\*kernel analysis is applied for extracting features from input images

**Method**

This report focuses on the comparative study of different values of K in KNN algorithm applied on at&t faces dataset using different normalization techniques. By applying a suitable facial recognition algorithm (namely KNN) to compare faces with the database. And comparing the accuracy of the predicted results from it.

K-Nearest Neighbors (KNN) is a supervised learning algorithm and can be used to solve classification and regression problems. It chooses the K closest neighbors and then based on these neighbors, assigns a class (for classification problems) or predicts a value (for regression problems) for a new observation. The K is the number of neighbors, which must be an odd number to avoid having equal votes. KNN uses Euclidean distance as the distance metric.



Euclidean distance:

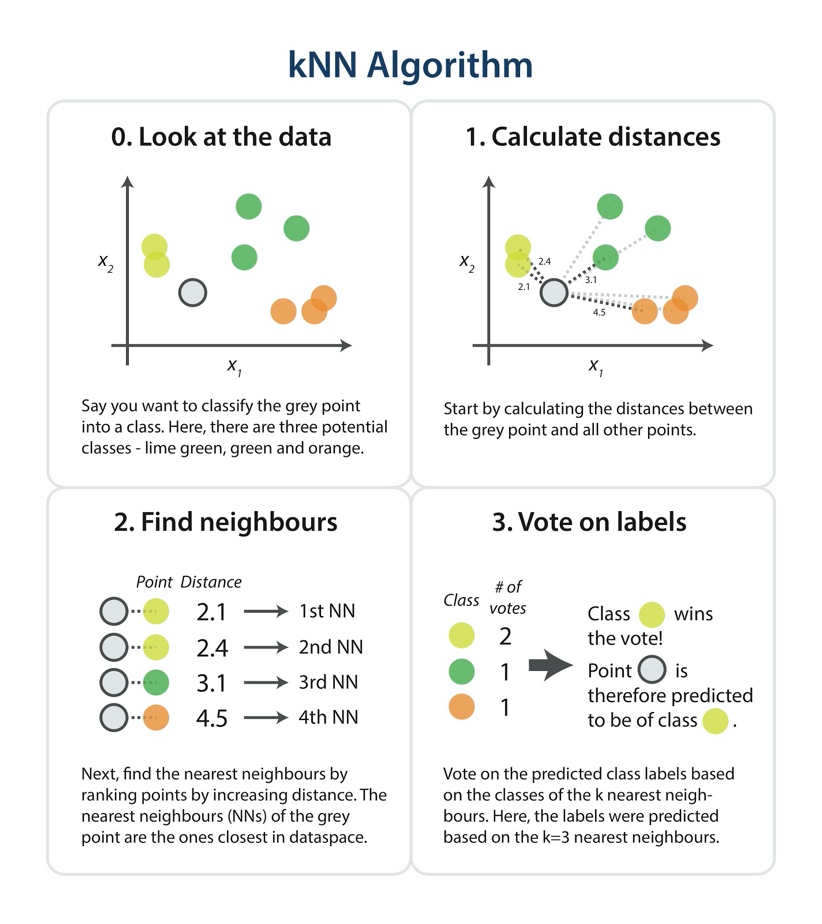
Normalization (﻿feature scaling) is putting the values in the same range or same scale so that no variable is dominated by the other.

To Normalize the data, ﻿different feature scaling techniques were used, listed below:

* **﻿Standarization**
  + Standardize features by removing the mean and scaling to unit variance. removing the average face (mean of pixels over the dataset) from each face.
* Min-Max Scaling
  + This technique scales the features to lie between a given minimum and maximum value, often between zero and one
* ﻿Binarizing
  + Binarizing assigns a Boolean value (*True*or *False*) to each sample based on a threshold. All values below or equal to the threshold are replaced by 0, above it by 1.
* ﻿Normalizing
  + Normalization is the process of scaling individual samples to have unit norm. So each sample with at least one non zero component is rescaled independently of other samples so that its norm (l1 or l2) equals one.
* Other: Principal Component Analysis [PCA]
  + PCA tries to get the features with maximum variance and the variance is high for high magnitude features.

|  |  |  |
| --- | --- | --- |
| Standardization | Min-Max Scaling | Normalizing |
|  |  | A close up of a clock  Description automatically generated |

the following steps will be followed:

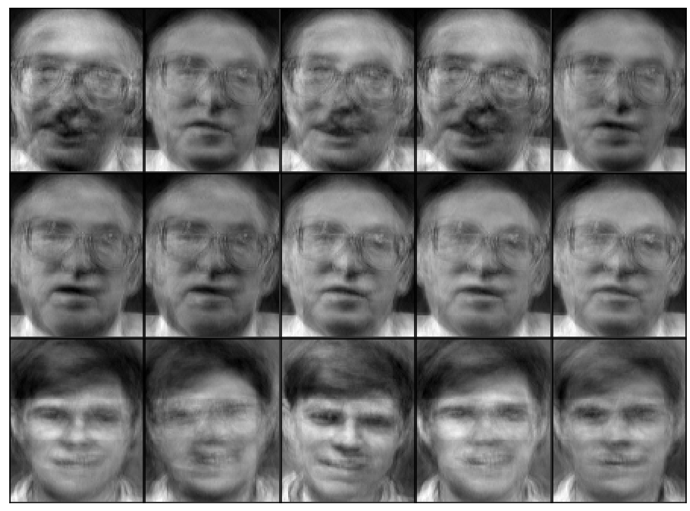
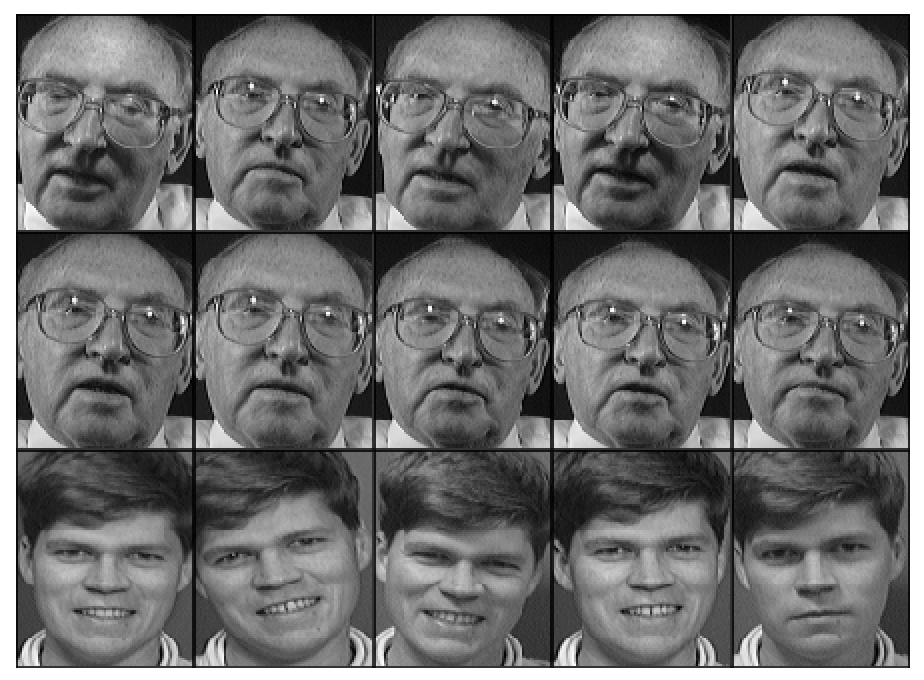
How KNN works:

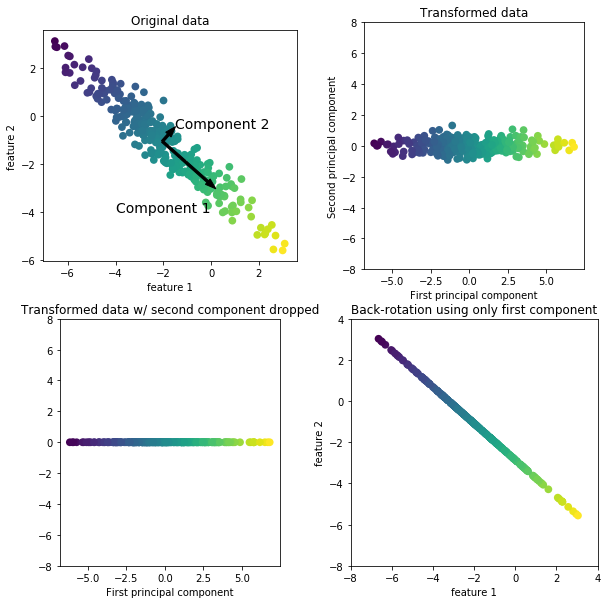
Some references:

* https://towardsdatascience.com/a-simple-introduction-to-k-nearest-neighbors-algorithm-b3519ed98e
* https://scikit-learn.org/stable/modules/preprocessing.html
* <https://medium.com/@moussadoumbia_90919/elbow-method-in-supervised-learning-optimal-k-value-99d425f229e7>
* **Principal Component Analysis [PCA]**

PCA method finds the underlying structure of the data in which there is the most variance by converting the images into a lowdimension space and perform a linear matrix transformation that finds the data variance in the projection subspace.

Set of mages before and after applying PCA:



A figure explaining how PCA works:

**Dataset**

AT&T Database of Faces.

The AT&T Database is a face images dataset that was made between 1992 and 1994 at AT&T Laboratories Cambridge. It contains 10 different images of 40 distinct people with 400 face images.

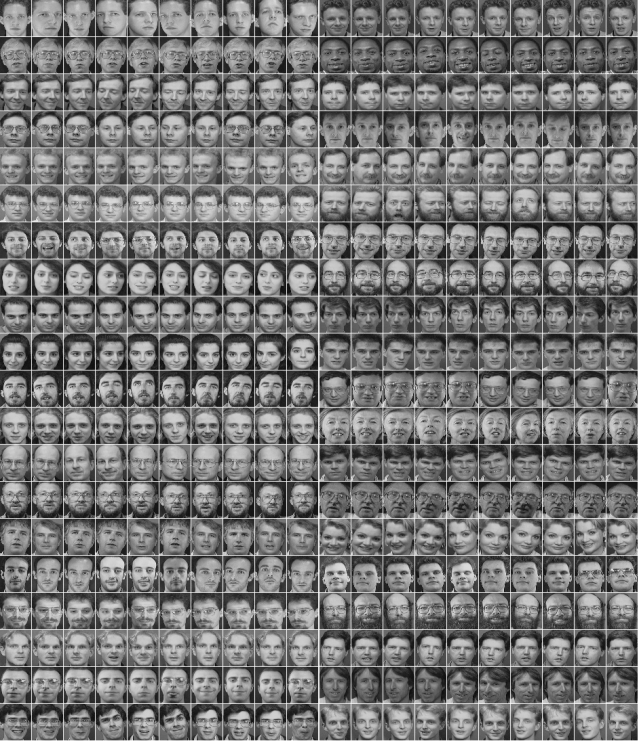
The images were captured at different conditions and times and different facial expressions. All of them has dark background. The format of the images is PGM. The size of each image is 92x112 pixels, with 256 grey levels per pixel.

The images are organized in 40 directories (one for each subject), which have names of the form sX, where X indicates the subject number (between 1 and 40). In each of these directories, there are ten different images of that subject, which have names of the form Y.pgm, where Y is the image number for that subject (between 1 and 10).

The dataset has 400 samples, 1024 features -which are the pixels-, 40 discrete classes from 1 to 10.

The processing of the data is descried in the Experiments –next page-.

Database link: <https://www.kaggle.com/kasikrit/att-database-of-faces#10.pgm>



**Experiments**

KNN Classifier was used for the classification process .

In the pre-processing, images were loaded with size ﻿48x48, ﻿converted from 2d image to 1d vector, ﻿and stored in one matrix of size 400x2304 ﻿( number of images(40x10) x number of features, pixels in image(48x48)).

Then the dataset was Split into Training and Testing sets: 70% or 7 images per subject as training and 30% or 3 images per subject as testing subsets.

The data was Normalized using ﻿different feature scaling techniques. (as described on the method).

The classifier was trained the on training set and tested on the testing set.

Finally, Compute the performance by comparing the real label (y\_test) to the labels predicted by the classifier (y\_pred). used accuracy\_score ﻿from sklearn.

We conducted some experiments to study and test the system performance as described below:

Analysis & Results:

1. KNN performance with different values of K.
   * To find the optimal value of K we tested KNN with different values of K from 1 to 40, calculated the Mean Error of KNN and plotted the results in a graph.
   * A screenshot of a map

     Description automatically generatedFrom the output we can see that the mean error is close to zero when the value of the K is between 1 and 5.
2. KNN performance with different feature scaling techniques.
   * Feature scaling techniques used:
     1. **Standarization**
     2. Min-Max Scaling
     3. ﻿Binarizing
     4. ﻿Normalizing
   * The comparison was done by calculating the performance Accuracy of KNN.
   * **From the results, it’s clear that scaling with binarizing isn’t efficient at all, in this example (the accuracy is less than 5% !). For the other three features the results are similar, though they change in every try (Range between ﻿85% and ﻿97%). We can say that the three** techniques (**Standarizati**on, Min-Max, Normalization) are equally efficient.

**A screenshot of a cell phone

Description automatically generated**

1. KNN performance vs K for each feature scaling techniques.
   * Finally all the techniques were tried with different K values ranging from 1 to 40.
   * The comparisons were done by calculating the Mean Error of KNN.
   * From the graph we can conclude that **Standarizati**on, Min-Max and Normalization are equally efficient except Binarizing. For K value the optimal values would be between 1 and 5.
   * Note: We can’t say which techniques has a performance higher than the other, because the results change in each run.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Mean Error of KNN in feature scaling techniques | | | |
| K value | Min-Max | **Standarizati**on | Normalization | Binarizing |
| 1 | 0.025 | 0.008 | 0.9833 | 0.0166 |
| 2 | 0.0916 | 0.066 | 0.9833 | 0.0166 |
| 3 | 0.0916 | 0.066 | 0.9833 | 0.0166 |
| 4 | 0.125 | 0.1 | 0.9833 | 0.06 |
| 5 | 0.1416 | 0.108 | 0.9833 | 0.083 |
| 6 | 0.216 | 0.125 | 0.9833 | 0.123 |
| 7 | 0.258 | 0.14 | 0.9833 | 0.175 |
| 8 | 0.275 | 0.175 | 0.9833 | 0.208 |
| 9 | 0.308 | 0.208 | 0.966 | 0.25 |
| 10 | 325 | 0.225 | 0.966 | 0.26 |

A screenshot of a map

Description automatically generated

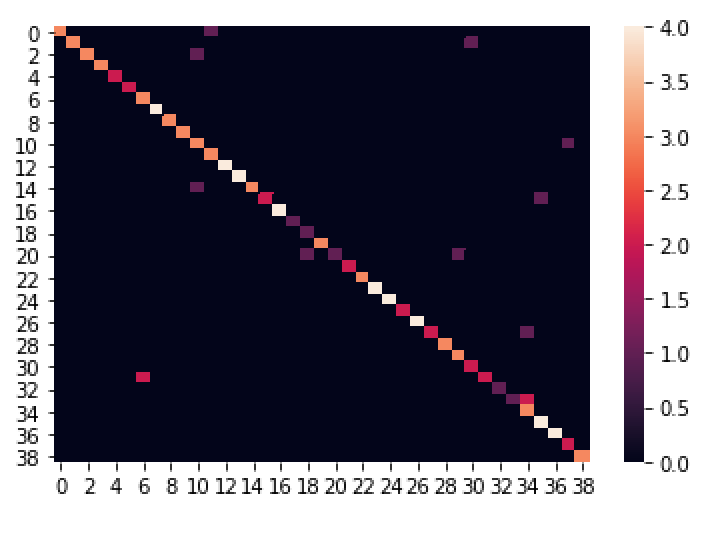
**Conclusion**

In this report we have studied how different Normalization Techniques, and different K values affect KNN performance in Face Recognition System.

The experimental results have proven that the proposed system can work with different normalization techniques. We have compared four normalization techniques, namely Min-Max, **Standarizati**on, Binarizing and Normalization with different values of K. The average accuracy came out to be 93 % for Min-Max 92 % for Normalization 93% for **Standarizati**on and 3 % for Binarizing. All the techniques had great similar performance except for Binarizing.

For K value the accuracy decreases as K increase, and the best value of K is between 1 and 5. Also, the results show that the system has a high recognition rate and the accuracy achieved up to 97%.

**More Data**

Confusion matrix graph

Classification report

