

Summer Training Report

On

Face Recognition System With Face Detection

Submitted in partial fulfilment of the requirement for
the degree of

Bachelor Of Technology In Computer Science

by

Vishal Yadav- 03096302717



Maharaja Surajmal Institute Of Technology

(Affiliated to Guru Gobind Singh Indraprastha University)

Janakpuri, New Delhi-58

DECLARATION

I **Vishal Yadav** (Enrollment number : 03096302717) currently in 5th semester pursuing Bachelor of technology in Computer Science from Maharaja Surajmal Institute Of Technology, New Delhi hereby declare that the “**Summer Training Report on Face recognition with Face Detection** “ is my original work and all the report breach no copyright or license , and are created by me on my own, or using resources that I have complete permission to use.

All the data given in this report is authentic to the best of my knowledge, and this report has not been submitted to any institute for the award of any other degree.

Vishal Yadav

03096302717

Maharaja Surajmal Institute Of Technology

September 2019

CERTIFICATE

VERIFIED CERTIFICATE OF COMPLETION



VISHAL YADAV

Has successfully completed the

Machine Learning

PROJECT BASED COURSE

Mudit Goel

Academic Head and Founder
Coding Elements



3rd June, 2019 to 28th July, 2019

www.codingelements.com

Certificate From Organisation

MAHARAJA SURAJMAL INSTITUTE OF TECHNOLOGY

Summer/Industrial Training Evaluation Form

F05 (MSIT-EXM-PA-02)

(Year 20..... – 20.....)

Details of the Student

Name.....

Roll No.....

Branch and Semester.....

Mobile No.....

E-mail ID.....

Details of the Organisation

Name and address of organization.....

Broaden Area.....

Name of Instructor.....

Designation and Contact No.....

Student Performance Record

| | No. of days Scheduled for the training | Number of days actually attended | Curriculum Scheduled for the student | Curriculum actually covered by the student |
|--------|--|---|--------------------------------------|---|
| Week 1 | | | | |
| Week 2 | | | | |
| Week 3 | | | | |
| Week 4 | | | | |
| Week 5 | | | | |
| Week 6 | | | | |

(Signature of the student)

Any comments or suggestions for the student performance during the training program (to be filled by
instructor).....

.....

(Signature of the instructor)

along with Seal

~~Note:~~ Every student has to fill and submit this Performa duly signed by his/her instructor to the faculty-
~~in-charge~~ by first week of September.

ACKNOWLEDGEMENT

The internship opportunity I had with Coding Elements was a great chance for learning and professional development. Therefore, I consider myself as a very lucky individual as I was provided with an opportunity to be a part of it. I am also grateful for having a chance to meet so many wonderful people and professionals who led me through this internship period.

Bearing in mind previous I am using this opportunity to express my deepest gratitude and special thanks to the Head of Coding Elements who in spite of being extraordinarily busy with his duties, took time out to hear, guide and keep me on the correct path and allowing me to carry out my project at their esteemed organization and extending during the training.

I express my deepest thanks to Mudit Goel , Head and founder of the Company for taking part in useful decision & giving necessary advice and guidance and arranged all facilities to make life easier. I choose this moment to acknowledge his contribution gratefully.

I perceive as this opportunity as a big milestone in my career development. I will strive to use gained skills and knowledge in the best possible way, and I will continue to work on their improvement, in order to attain desired career objectives. Hope to continue cooperate with all of you in the future,

Sincerely,

Vishal Yadav

Place: New Delhi

Date: June 2019 - July 2019

FACE DETECTION SYSTEM WITH FACE RECOGNITION

ABSTRACT

The face is one of the easiest ways to distinguish the individual identity of each other. Face recognition is a personal identification system that uses personal characteristics of a person to identify the person's identity. Human face recognition procedure basically consists of two phases, namely face detection, where this process takes place very rapidly in humans, except under conditions where the object is located at a short distance away, the next is the introduction, which recognize a face as individuals. Stage is then replicated and developed as a model for facial image recognition (face recognition) is one of the much-studied biometrics technology and developed by experts.

There are two kinds of methods that are currently popular in developed face recognition pattern namely, Eigenface method and Fisherface method. Facial image recognition Eigenface method is based on the reduction of face dimensional space using Principal Component Analysis (PCA) for facial features. The main purpose of the use of PCA on face recognition using Eigen faces was formed (face space) by finding the eigenvector corresponding to the largest eigenvalue of the face image. The area of this project face detection system with face recognition is Image processing. The software requirements for this project is Jupyter Notebook software. This report describes the face detection and recognition mini-project undertaken for visual perception. It reports the technologies available in the Open-Computer-Vision(Open CV) library and methodology to implement them using Python. For face detection, Haar-Cascades were used and for face recognition Eigenfaces, Fisherfaces and Local binary pattern histograms were used.

The methodology is described including flow charts for each stage of the system. Next, the results are shown including plots and screen-shots followed by a discussion of encountered challenges.

Extension: There are vast number of applications from this face detection project, this project can be extended that the various parts in the face can be detect which are in various directions and shapes. Next, the results are shown include screen-shots.

INTRODUCTION

The following project involve face recognition and face detection on the basic of dataset. First the dataset is generated which include faces of different persons and then when ever a face come in front of camera face detection techniques works and classify the face of the person. It involved building a system for face detection and face recognition using several classifiers available in the open computer vision library (OpenCV). Face recognition is a non-invasive identification system and faster than other systems since multiple faces can be analysed at the same time. The difference between face detection and identification is, face detection is to identify a face from an image and locate the face. The project basically classify images on the un-supervised algorithm known as KNN- K nearest neighbour.

Face recognition is making the decision "whose face is it ? ", using an image database. In this project both are accomplished using different techniques and are described below. The report begins with a brief history of face recognition. This is followed by the explanation of HAAR-cascades, Eigenface, Fisherface and Local binary pattern histogram (LBPH) algorithms. Next, the methodology and the results of the project are described. A discussion regarding the challenges and the resolutions are described. Finally, a conclusion is provided on the pros and cons of each algorithm and possible implementations.

INDEX

| Contents | Page Number |
|--|-------------|
| Certificate..... | 3 |
| Acknowledgement..... | 5 |
| Abstract..... | 6 |
| Introduction..... | 7 |
| Company Profile..... | 9 |
| Technology tools used..... | 10 |
| Numpy..... | 10 |
| Pandas..... | 12 |
| Open CV..... | 14 |
| Python OS Module..... | 16 |
| Haar-Cascade..... | 17 |
| Cascade-Classfier..... | 19 |
| Sklearn..... | 21 |
| Unsupervised Learning..... | 23 |
| Types of unsupervised learning..... | 24 |
| Clustering Types..... | 24 |
| KNN..... | 29 |
| Face Recognition System..... | 33 |
| Application of face recognition system..... | 34 |
| Code of project..... | 42 |
| Demonstration of technology through project..... | 48 |
| Conclusion..... | 55 |
| Bibliography..... | 56 |
| Appendices..... | 58 |

COMPANY PROFILE

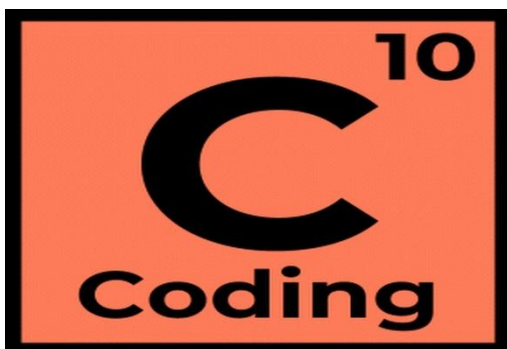
- Coding Elements
- Established in [2018]
- Physical address per location [H-10 Express Arcade Netaji Subhash Place Suite 106, Netaji Subhash Place, Wazirpur, Delhi, 110034]
- Email_id info@codingelements.com
- Phone No 9999643211

About Us / Our Story

Coding Elements is a coding based training company founded by Mudit Goel in 2018. It is situated in Netaji Subhash Palace. Faculty placed there are highly experienced and have worked in the US. While some have worked on big platforms like IBM, Amazon and many more. It is a place where new enthusiast and creative coders can make an experience with many programming languages like Python, C++, Machine Learning. It helps us to think outside the box and come up with new innovative ideas which can make a better future.

Our Mission / Values

Their main motto or mission is to make a better coder so we can get the best out of our self and above all a better citizen as we are the future generation. Everybody in this country should learn to program a computer, because it teaches you how to think.



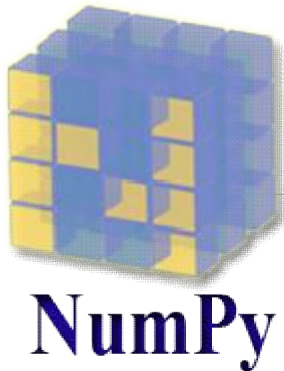
TECHNOLOGY TOOLS USED

NumPy

NumPy is a module for Python. The name is an acronym for "Numeric Python" or "Numerical Python". It is an extension module for Python, mostly written in C. This makes sure that the precompiled mathematical and numerical functions and functionalities of Numpy guarantee great execution speed.

Furthermore, NumPy enriches the programming language Python with powerful data structures, implementing multi-dimensional arrays and matrices. These data structures guarantee efficient calculations with matrices and arrays. The implementation is even aiming at huge matrices and arrays, better known under the heading of "big data". Besides that the module supplies a large library of high-level mathematical functions to operate on these matrices and arrays.

SciPy (Scientific Python) is often mentioned in the same breath with NumPy. SciPy needs Numpy, as it is based on the data structures of Numpy and furthermore its basic creation and manipulation functions. It extends the capabilities of NumPy with further useful functions for minimization, regression, Fourier-transformation and many others.



Both NumPy and SciPy are not part of a basic Python installation. They have to be installed after the Python installation. NumPy has to be installed before installing SciPy.

NumPy is based on two earlier Python modules dealing with arrays. One of these is Numeric. Numeric is like NumPy Python module for high-performance, numerical computing, but it is obsolete nowadays. Another predecessor of NumPy is Numarray, which is a complete rewrite of Numeric but is deprecated as well. NumPy is a merger of those two, i.e. it is build on the code of Numeric and the features of Numarray.

It is the fundamental package for scientific computing with Python. It contains various featuresincluding these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data.

Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

Installation:

- Mac and Linux users can install NumPy via pip command: `pip install numpy`
- For Windows same command can be used with command prompt for this you need to first install the pip command and the python installation manager.

Comparison between Core python and NumPy

When we say "Core Python", we mean Python without any special modules, i.e. especially without NumPy.

The advantages of Core Python:

- high-level number objects: integers, floating point
 - containers: lists with cheap insertion and append methods, dictionaries with fast lookup
- Advantages of using Numpy with Python:

- array oriented computing
- efficiently implemented multi-dimensional arrays
- designed for scientific computation

Why is NumPy fast

Vectorization describes the absence of any explicit looping, indexing, etc, in the code - these things are taking place, of course, just “behind the scenes” in optimized, pre-compiled C code.

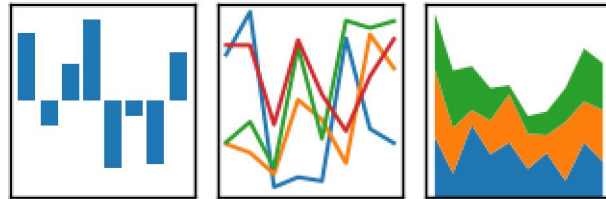
Vectorized code has many advantages, among which are:

- vectorized code is more concise and easier to read
- fewer lines of code generally means fewer bugs
- the code more closely resembles standard mathematical notation (making it easier, typically, to correctly code mathematical constructs)
- vectorization results in more “Pythonic” code. Without vectorization, our code would be littered with inefficient and difficult to read for loops.

PANDAS

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



In computer programming, **pandas** is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals.

Library features

- Data Frame object for data manipulation with integrated indexing.
- Tools for reading and writing data between in-memory data structures and different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and sub-setting of large data sets.
- Data structure column insertion and deletion.
- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.
- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
- Time series-functionality: Date range generation^[4] and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging.
- Provides data filtration

Installing Pandas

```
! pip install pandas
```

Import Pandas

In order to import Pandas all you have to do is run the following code:

```
import pandas as pd  
import numpy as np
```

OPEN CV



OpenCV is a Python library which is designed to solve computer vision problems. OpenCV was originally developed in 1999 by Intel but later it was supported by Willow Garage.

OpenCV supports a wide variety of programming languages such as C++, Python, Java etc. Support for multiple platforms including Windows, Linux, and MacOS.

OpenCV Python is nothing but a wrapper class for the original C++ library to be used with Python. Using this, all of the OpenCV array structures gets converted to/from NumPy arrays.

This makes it easier to integrate it with other libraries which use NumPy. For example, libraries such as SciPy and Matplotlib.

Loading an image using OpenCV:

```
1  Import cv2
2
3
4
5  # colored Image
6
7  Img = cv2.imread ("Penguins.jpg",1)
8
9
10
11 # Black and White (gray scale)
12
13 Img_1 = cv2.imread ("Penguins.jpg",0)
```

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies.

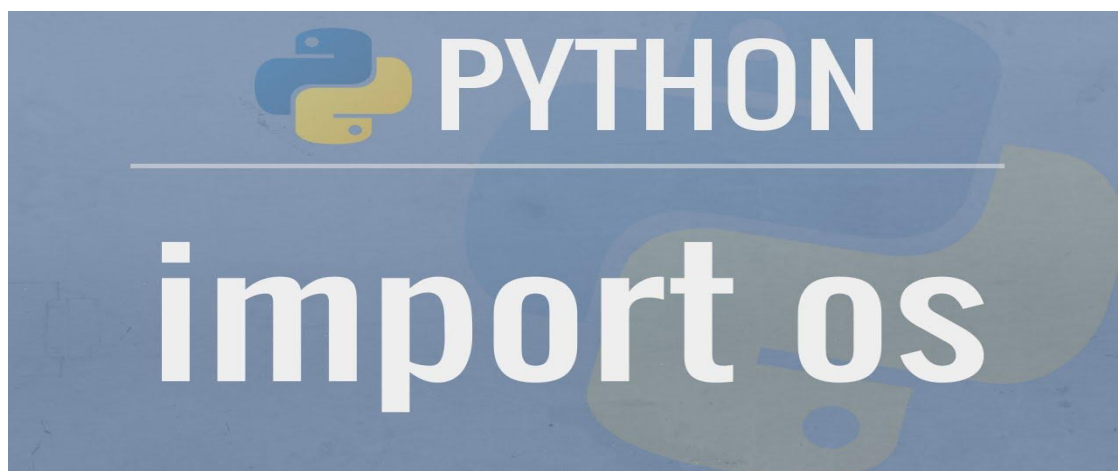
Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many startups such as Applied Minds, VideoSurf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses span the range from stitching streetview images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured CUDA and OpenCL interfaces are being actively developed right now. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

OpenCV has a modular structure, which means that the package includes several shared or static libraries. The following modules are available:

- **Core functionality (core)** - a compact module defining basic data structures, including the dense multi-dimensional array Mat and basic functions used by all other modules.
- **Image Processing (imgproc)** - an image processing module that includes linear and non-linear image filtering, geometrical image transformations (resize, affine and perspective warping, generic table-based remapping), color space conversion, histograms, and so on.
- **Video Analysis (video)** - a video analysis module that includes motion estimation, background subtraction, and object tracking algorithms.
- **Camera Calibration and 3D Reconstruction (calib3d)** - basic multiple-view geometry algorithms, single and stereo camera calibration, object pose estimation, stereo correspondence algorithms, and elements of 3D reconstruction.
- **2D Features Framework (features2d)** - salient feature detectors, descriptors, and descriptor matchers.
- **Object Detection (objdetect)** - detection of objects and instances of the predefined classes (for example, faces, eyes, mugs, people, cars, and so on).
- **High-level GUI (highgui)** - an easy-to-use interface to simple UI capabilities.
- **Video I/O (videoio)** - an easy-to-use interface to video capturing and video codecs.
- ... some other helper modules, such as FLANN and Google test wrappers, Python bindings, and others.

Python OS Module



The OS module in Python provides a way of using operating system dependent functionality.

The functions that the OS module provides allows you to interface with the underlying operating system that Python is running on- be that Windows, Mac or linux.

OS Functions

import OS

Executing a shell command- `os.system()`

Get the users environment-`os.environ()`

Returns the current working directory- `os.getcwd()`

Return the real group id of the current process-`os.getgid()`

Return the current process's ID of the current process- `os.getpid()`

Set the current numeric umask and return the previous u mask-`os.umask(mask)`

Return information identifying the current operating system-`os.uname()`

Change the root directory of the current process to path-`os.chroot(path)`

Return a list of the entries in the directory given by path-`os.listdir(path)`

Create a directory named path with numeric mode –`os.mkdir(path)`

Recursive directory creation function-`os.makedirs(path)`

Remove (delete) the file path- `os.remove(path)`

Remove directories recursively-`os.removedirs(path)`

Rename the file or directory src to dst-`os.rename(src,dst)`

Remove(delete) the directory path-`os.rmdir(path)`

HAAR-CASCADE



Haar Cascade is a machine learning object detection algorithm used to identify objects in an image or video and based on the concept of features proposed by Paul Viola and Michael Jones in their paper "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001.

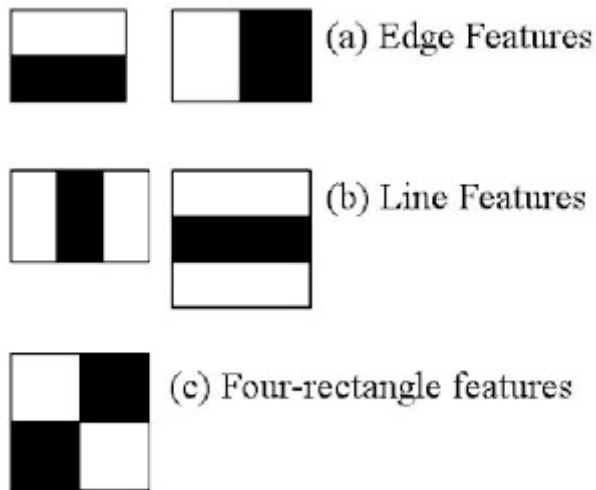
It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. The algorithm has four stages:

1. **Haar** Feature Selection
2. Creating **Integral Images**
3. **Adaboost** Training
4. Cascading Classifiers

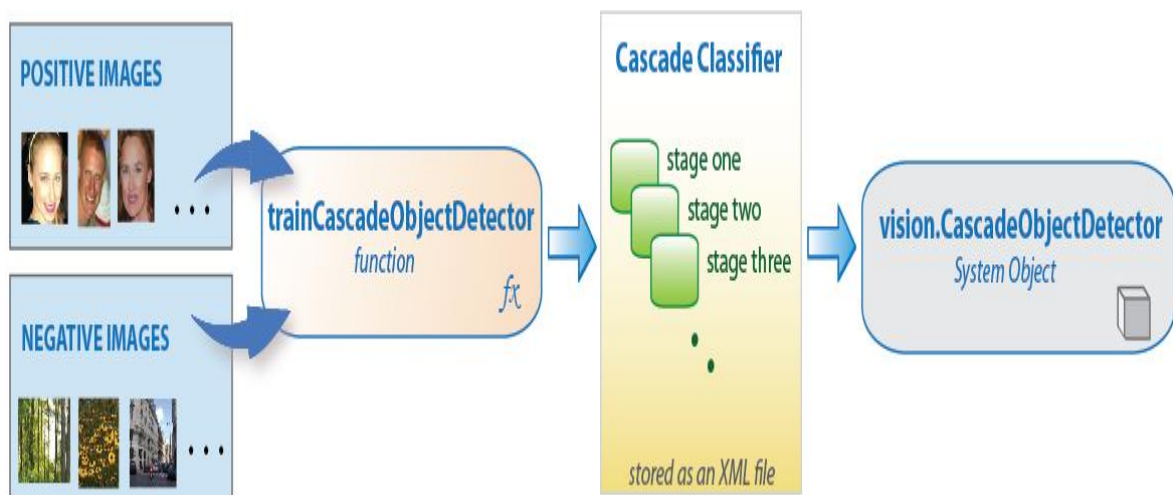
It is well known for being able to detect faces and body parts in an image, but can be trained to identify almost any object.

Lets take face detection as an example. Initially, the **algorithm needs a lot of positive images of faces and negative images** without faces to train the classifier. Then we need to extract features from it.

First step is to collect the **Haar** Features. A **Haar** feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums.



CASCADE_CLASSIFIER



The cascade classifier consists of a collection of stages, where each stage is an ensemble of weak learners. The weak learners are simple classifiers called *decision stumps*. Each stage is trained using a technique called boosting. *Boosting* provides the ability to train a highly accurate classifier by taking a weighted average of the decisions made by the weak learners.

Each stage of the classifier labels the region defined by the current location of the sliding window as either positive or negative. *Positive* indicates that an object was found and *negative* indicates no objects were found. If the label is negative, the classification of this region is complete, and the detector slides the window to the next location. If the label is positive, the classifier passes the region to the next stage. The detector reports an object found at the current window location when the final stage classifies the region as positive.

The stages are designed to reject negative samples as fast as possible. The assumption is that the vast majority of windows do not contain the object of interest. Conversely, true positives are rare and worth taking the time to verify.

- A *true positive* occurs when a positive sample is correctly classified.
- A *false positive* occurs when a negative sample is mistakenly classified as positive.
- A *false negative* occurs when a positive sample is mistakenly classified as negative.

To work well, each stage in the cascade must have a low false negative rate. If a stage incorrectly labels an object as negative, the classification stops, and you

cannot correct the mistake. However, each stage can have a high false positive rate. Even if the detector incorrectly labels a nonobject as positive, you can correct the mistake in subsequent stages. Adding more stages reduces the overall false positive rate, but it also reduces the overall true positive rate.

Cascade classifier training requires a set of positive samples and a set of negative images. You must provide a set of positive images with regions of interest specified to be used as positive samples. You can use the Image Labeler to label objects of interest with bounding boxes. The Image Labeler outputs a table to use for positive samples. You also must provide a set of negative images from which the function generates negative samples automatically. To achieve acceptable detector accuracy, set the number of stages, feature type, and other function parameters.

SKLEARN




Scikit-learn (formerly **scikits.learn**) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. Scikit-learn is largely written in Python, with some core algorithms written in Cython to achieve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR.

Some popular groups of models provided by sklearn includes:

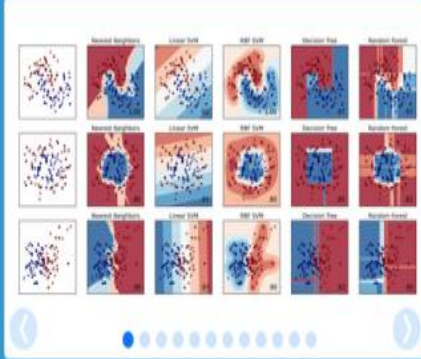
- **Clustering**: for grouping unlabeled data such as KMeans.
- **Cross Validation**: for estimating the performance of supervised models on unseen data.
- **Datasets**: for test datasets and for generating datasets with specific properties for investigating model behavior.
- **Dimensionality Reduction**: for reducing the number of attributes in data for summarization, visualization and feature selection such as Principal component analysis.
- **Ensemble methods**: for combining the predictions of multiple supervised models.
- **Feature extraction**: for defining attributes in image and text data.

- **Feature selection:** for identifying meaningful attributes from which to create supervised models.
- **Parameter Tuning:** for getting the most out of supervised models.
- **Manifold Learning:** For summarizing and depicting complex multi-dimensional data.
- **Supervised Models:** a vast array not limited to generalized linear models, discriminate analysis, naive bayes, lazy methods, neural networks, support vector machines and decision trees.



[Home](#)
[Installation](#)
[Documentation](#)
[Examples](#)

Fork me on GitHub



scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which set of categories a new observation belong to.

Applications: Spam detection, Image recognition.

Algorithms: SVM, nearest neighbors, random forest, ...

— Examples

Regression

Predicting a continuous value for a new example.

Applications: Drug response, Stock prices.

Algorithms: SVR, ridge regression, Lasso, ...

— Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: k-Means, spectral clustering, mean-shift, ...

— Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Algorithms: PCA, Isomap, non-negative matrix factorization.

— Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning

Modules: grid search, cross validation, metrics.

— Examples

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.

Modules: preprocessing, feature extraction.

— Examples

UNSUPERVISED LEARNING

Unsupervised learning is the training of machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data.

Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore machine is restricted to find the hidden structure in unlabeled data by our-self.

For instance, suppose it is given an image having both dogs and cats which have not seen ever.

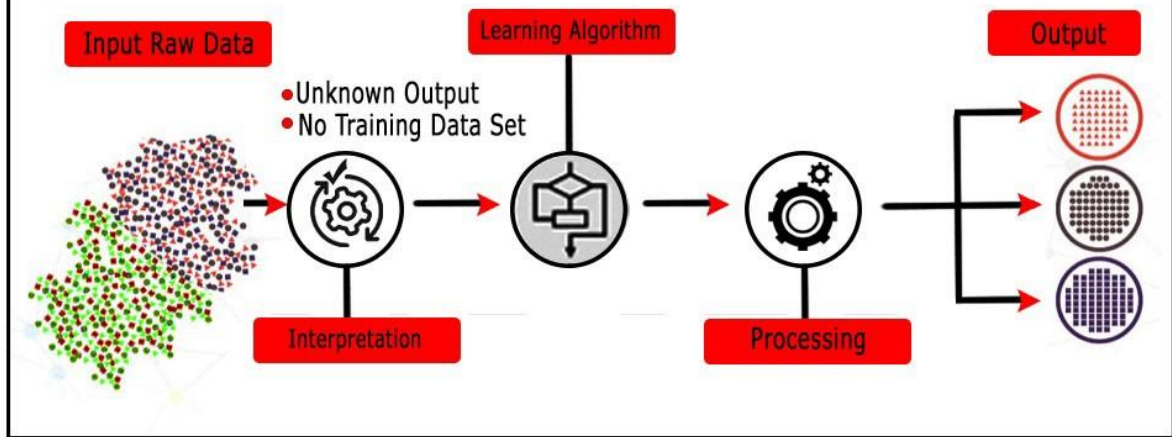


Thus the machine has no idea about the features of dogs and cat so we can't categorize it in dogs and cats. But it can categorize them according to their similarities, patterns, and differences i.e., we can easily categorize the above picture into two parts. First part may contain all pics having **dogs** in it and second part may contain all pics having **cats** in it. Here you didn't learn anything before, means no training data or examples.

Unsupervised learning classified into two categories of algorithms:

- **Clustering:** A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
- **Association:** An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

Unsupervised Learning



Types of Unsupervised Learning

Unsupervised learning problems further grouped into clustering and association problems.

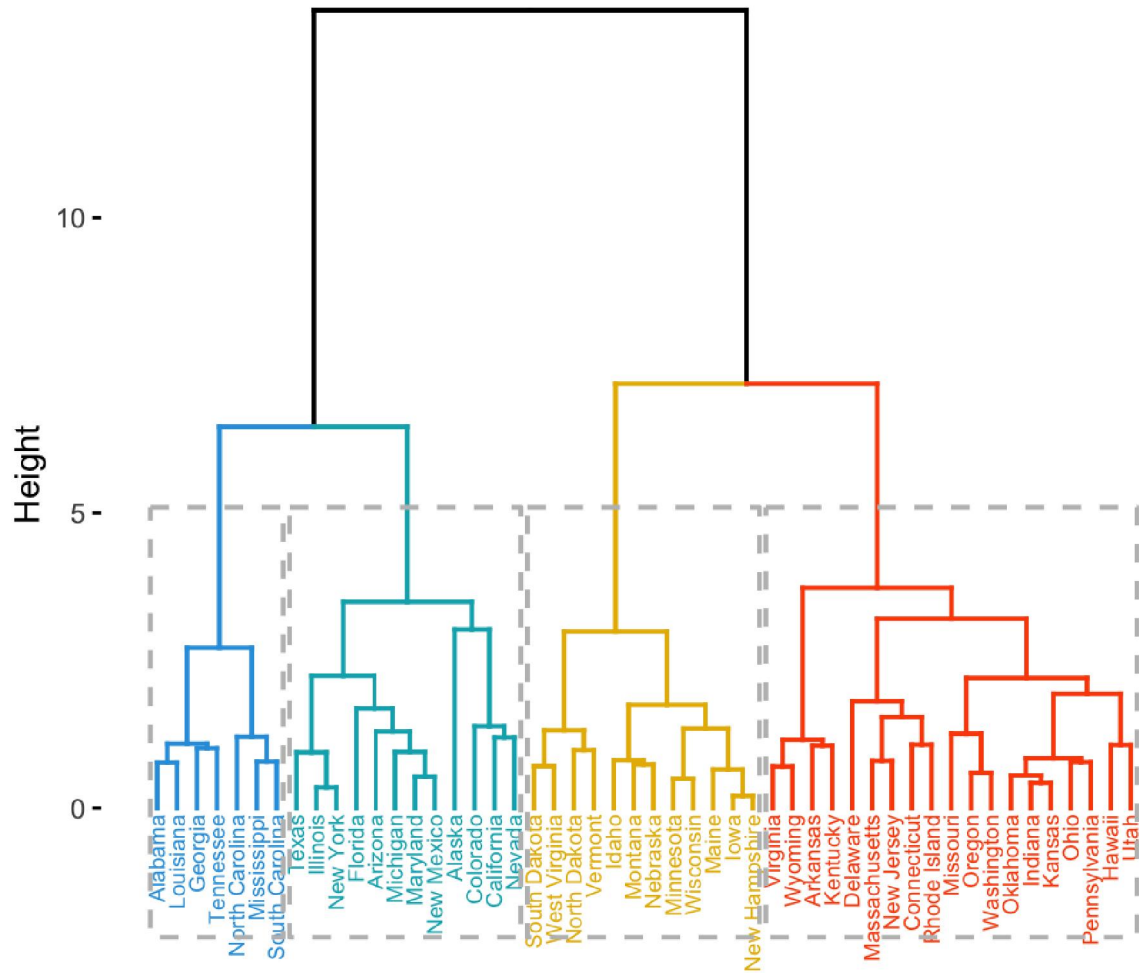
Clustering Types

- Hierarchical clustering
- K-means clustering
- K-NN (k nearest neighbors)
- Principal Component Analysis
- Singular Value Decomposition
- Independent Component Analysis

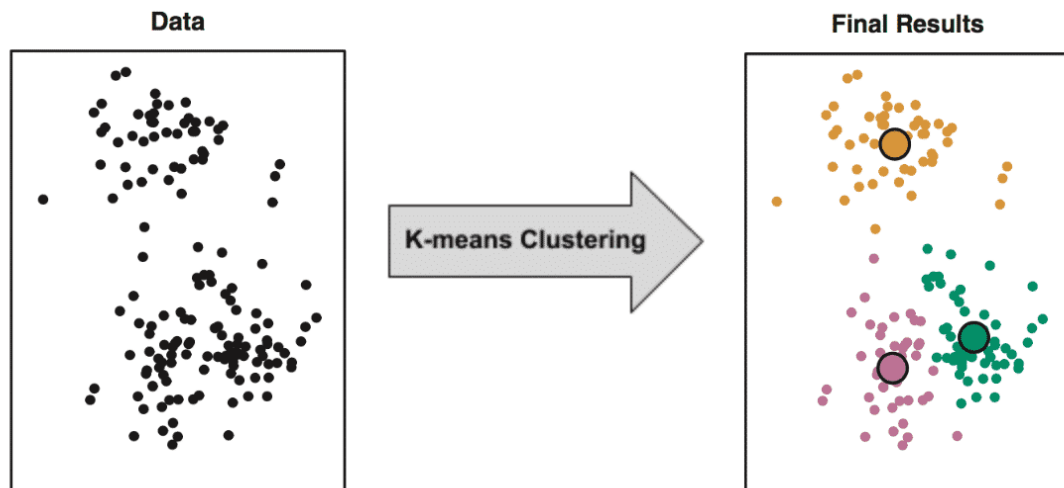
Hierarchical Clustering:

Hierarchical clustering is an algorithm which builds a hierarchy of clusters. It begins with all the data which is assigned to a cluster of their own. Here, two close cluster are going to be in the same cluster. This algorithm ends when there is only one cluster left.

Cluster Dendrogram



K-means Clustering



K means it is an iterative clustering algorithm which helps you to find the highest value for every iteration. Initially, the desired number of clusters are selected. In this clustering method, you need to cluster the data points into k groups. A larger k means smaller groups with more granularity in the same way. A lower k means larger groups with less granularity.

The output of the algorithm is a group of "labels." It assigns data point to one of the k groups. In k -means clustering, each group is defined by creating a centroid for each group. The centroids are like the heart of the cluster, which captures the points closest to them and adds them to the cluster.

K-mean clustering further defines two subgroups:

- Agglomerative clustering
- Dendrogram

Agglomerative clustering:

This type of K-means clustering starts with a fixed number of clusters. It allocates all data into the exact number of clusters. This clustering method does not require the number of clusters K as an input. Agglomeration process starts by forming each data as a single cluster.

This method uses some distance measure, reduces the number of clusters (one in each iteration) by merging process. Lastly, we have one big cluster that contains all the objects.

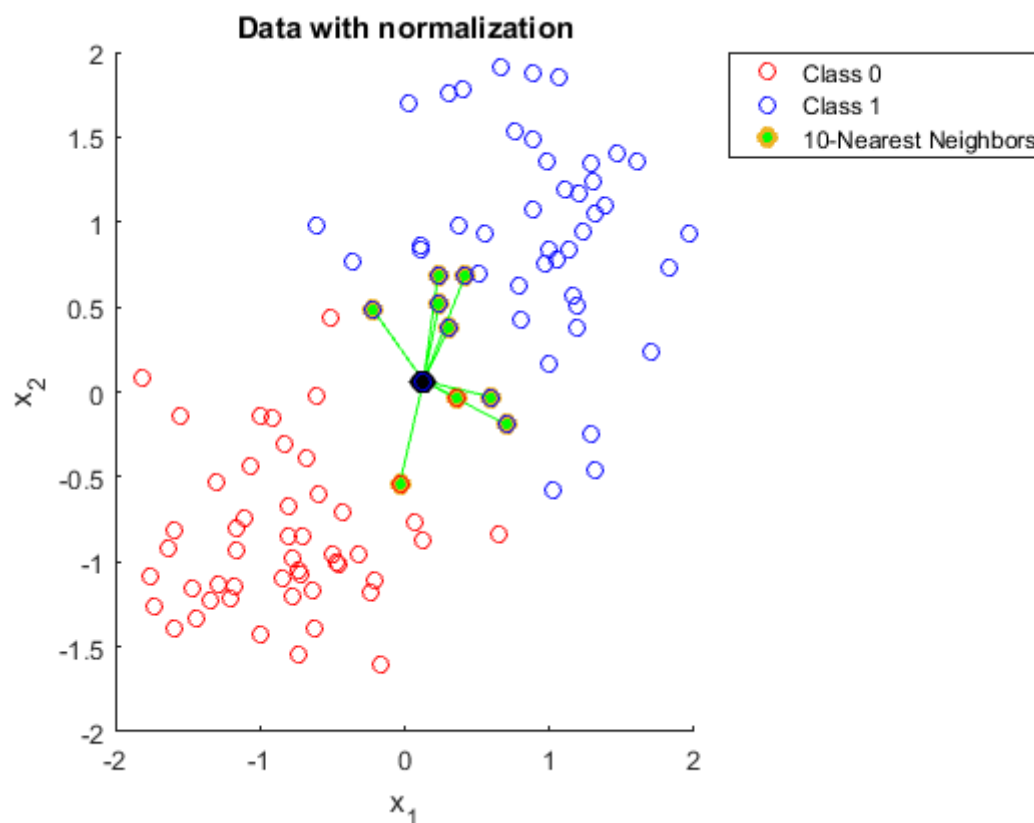
Dendrogram:

In the Dendrogram clustering method, each level will represent a possible cluster.

The height of dendrogram shows the level of similarity between two join clusters.

The closer to the bottom of the process they are more similar cluster which is finding of the group from dendrogram which is not natural and mostly subjective.

K- Nearest neighbours



K- nearest neighbour is the simplest of all machine learning classifiers. It differs from other machine learning techniques, in that it doesn't produce a model. It is a simple algorithm which stores all available cases and classifies new instances based on a similarity measure.

It works very well when there is a distance between examples. The learning speed is slow when the training set is large, and the distance calculation is nontrivial.

Principal Components Analysis:

In case you want a higher-dimensional space. You need to select a basis for that space and only the 200 most important scores of that basis. This base is known as a principal component. The subset you select constitute is a new space which is small in size compared to original space. It maintains as much of the complexity of data as possible.

Association

Association rules allow you to establish associations amongst data objects inside large databases. This unsupervised technique is about discovering interesting relationships between variables in large databases. For example, people that buy a new home most likely to buy new furniture.

Other Examples:

- A subgroup of cancer patients grouped by their gene expression measurements
- Groups of shopper based on their browsing and purchasing histories
- Movie group by the rating given by movies viewers

Supervised vs. Unsupervised Machine Learning

| Parameters | Supervised machine learning technique | Unsupervised machine learning technique |
|---------------|--|--|
| Input Data | Algorithms are trained using labeled data. | Algorithms are used against data which is not labelled |
| Computational | Supervised learning is a simpler | Unsupervised learning is computationally |

| Complexity | method. | complex |
|------------|---|---------------------------------------|
| Accuracy | Highly accurate and trustworthy method. | Less accurate and trustworthy method. |

Applications of unsupervised machine learning

Some applications of unsupervised machine learning techniques are:

- Clustering automatically split the dataset into groups base on their similarities
- Anomaly detection can discover unusual data points in your dataset. It is useful for finding fraudulent transactions
- Association mining identifies sets of items which often occur together in your dataset
- Latent variable models are widely used for data pre-processing. Like reducing the number of features in a dataset or decomposing the dataset into multiple components

Disadvantages of Unsupervised Learning

- You cannot get precise information regarding data sorting, and the output as data used in unsupervised learning is labeled and not known
- Less accuracy of the results is because the input data is not known and not labeled by people in advance. This means that the machine requires to do this itself.
- The spectral classes do not always correspond to informational classes.
- The user needs to spend time interpreting and label the classes which follow that classification.
- Spectral properties of classes can also change over time so you can't have the same class information while moving from one image to another.

KNN

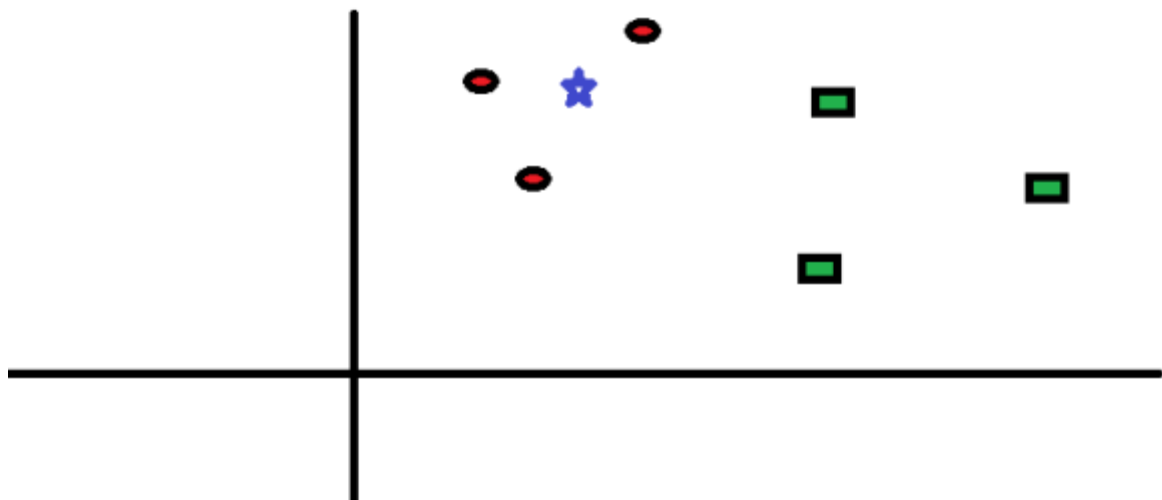
When do we use KNN algorithm?

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry. To evaluate any technique we generally look at 3 important aspects:

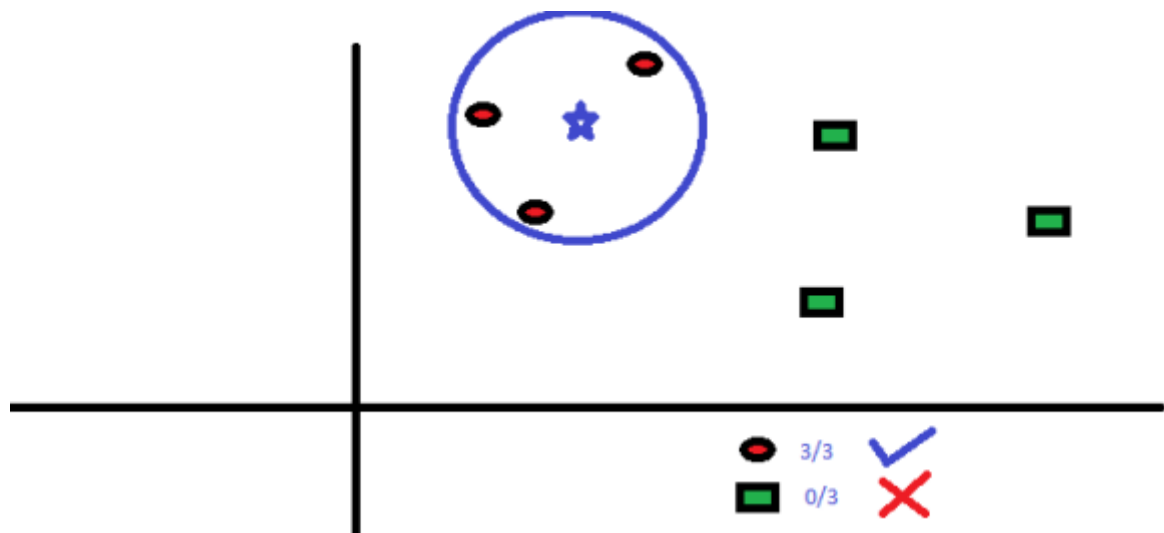
1. Ease to interpret output
2. Calculation time
3. Predictive Power

How does the KNN algorithm work?

Let's take a simple case to understand this algorithm. Following is a spread of red circles (RC) and green squares (GS) :



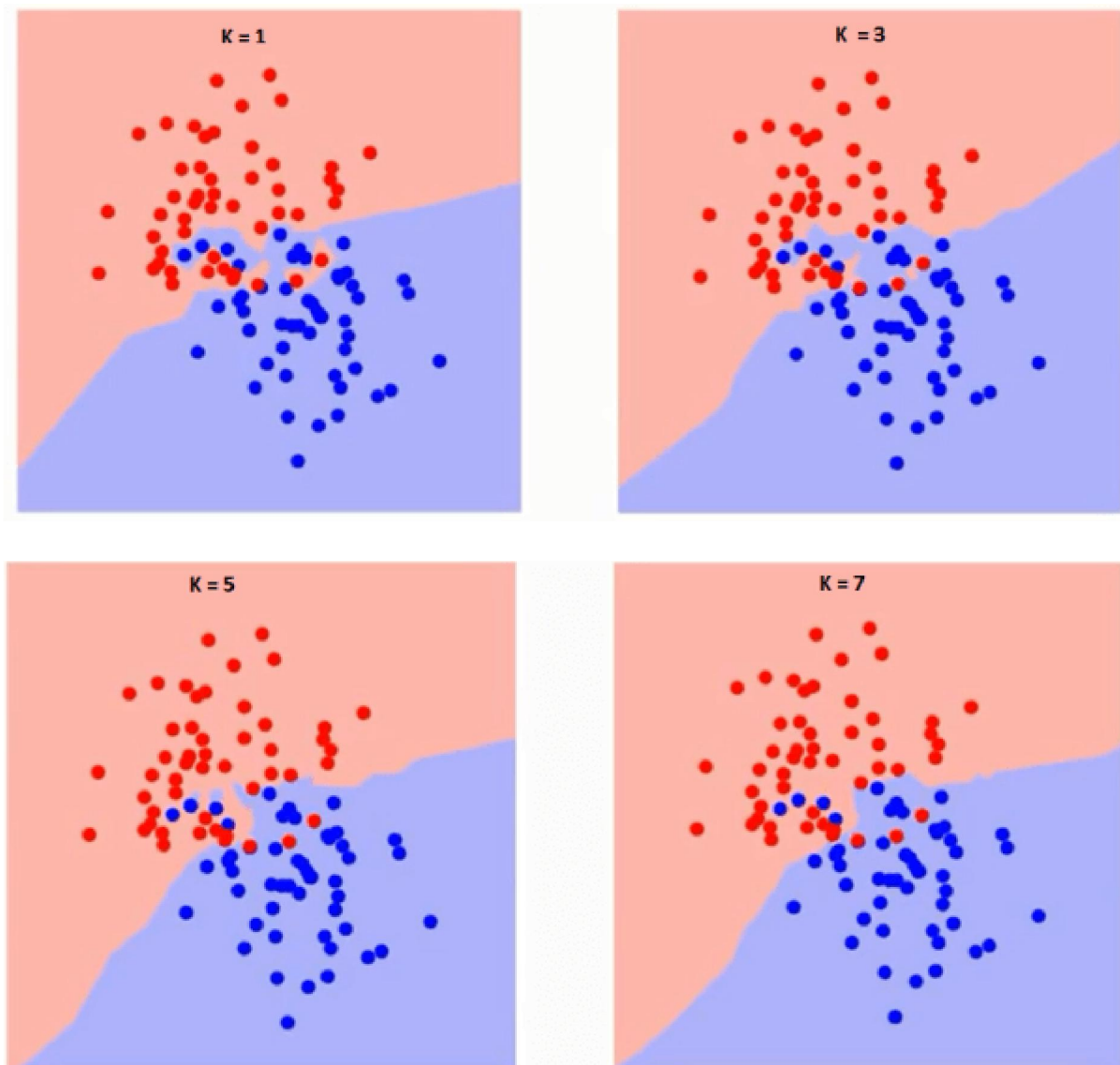
You intend to find out the class of the blue star (BS) . BS can either be RC or GS and nothing else. The “K” in KNN algorithm is the nearest neighbors we wish to take vote from. Let's say $K = 3$. Hence, we will now make a circle with BS as center just as big as to enclose only three datapoints on the plane. Refer to following diagram for more details:



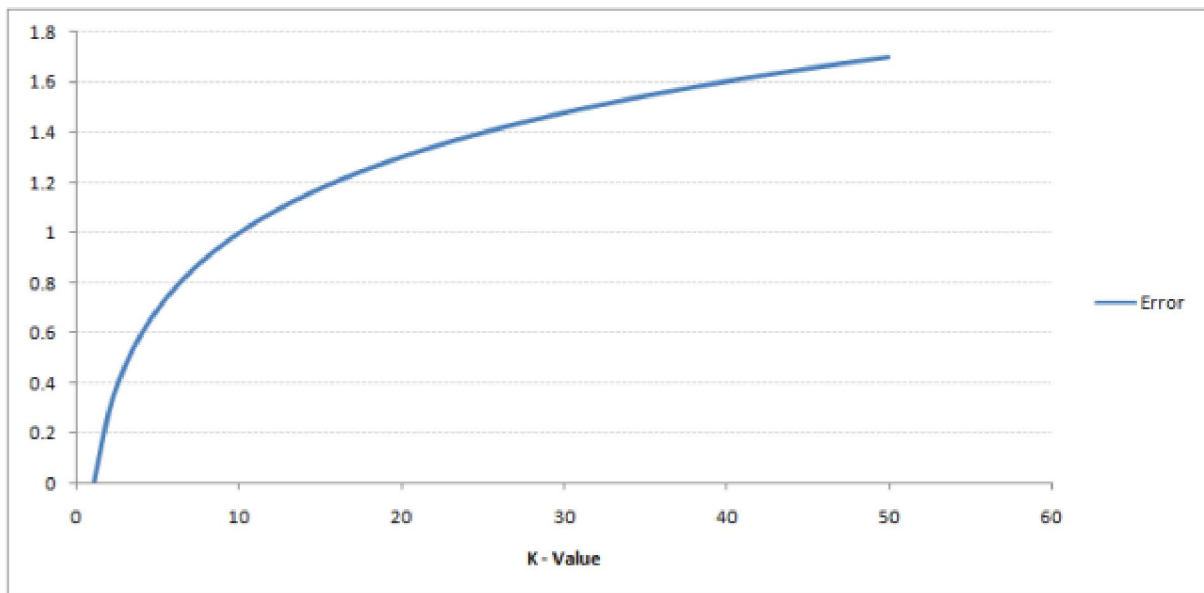
The three closest points to BS are all RC. Hence, with good confidence level we can say that the BS should belong to the class RC. Here, the choice became very obvious as all three votes from the closest neighbor went to RC. The choice of the parameter K is very crucial in this algorithm. Next we will understand what are the factors to be considered to conclude the best K .

How do we choose the factor K ?

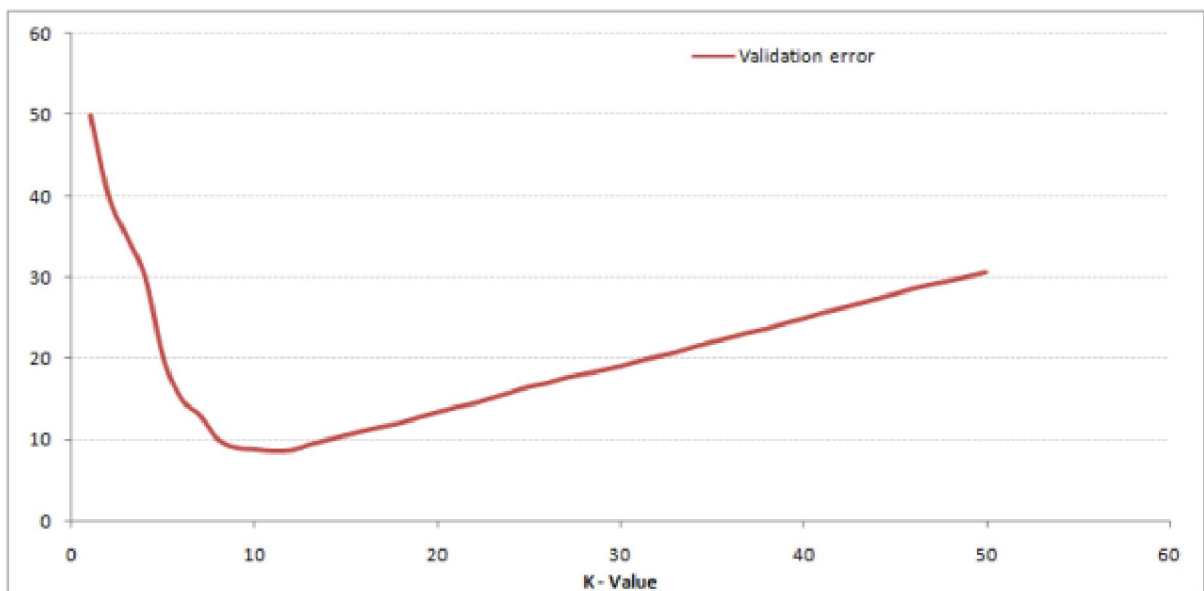
First let us try to understand what exactly does K influence in the algorithm. If we see the last example, given that all the 6 training observations remain constant, with a given K value we can make boundaries of each class. These boundaries will segregate RC from GS. The same way, let's try to see the effect of value " K " on the class boundaries. Following are the different boundaries separating the two classes with different values of K .



If you watch carefully, you can see that the boundary becomes smoother with increasing value of K. With K increasing to infinity it finally becomes all blue or all red depending on the total majority. The training error rate and the validation error rate are two parameters we need to access on different K-value. Following is the curve for the training error rate with varying value of K :



As you can see, the error rate at $K=1$ is always zero for the training sample. This is because the closest point to any training data point is itself. Hence the prediction is always accurate with $K=1$. If validation error curve would have been similar, our choice of K would have been 1. Following is the validation error curve with varying value of K :

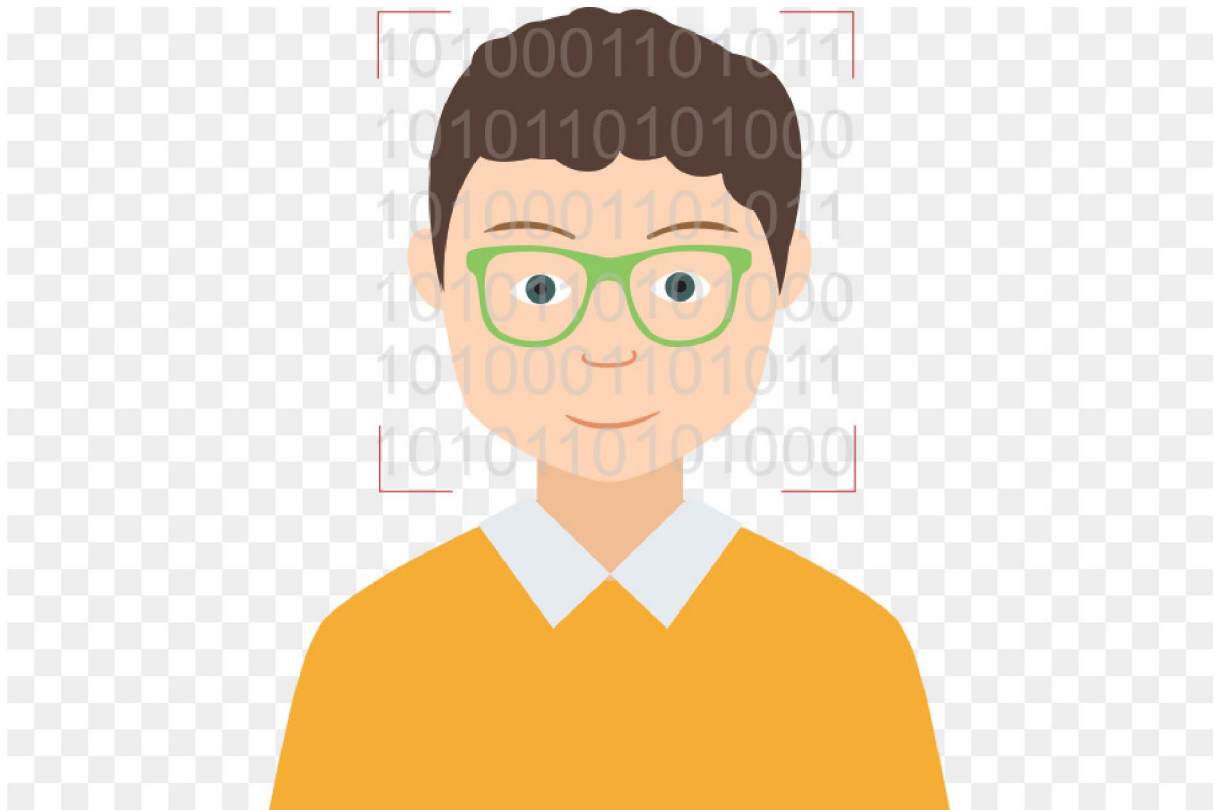


This makes the story more clear. At $K=1$, we were overfitting the boundaries. Hence, error rate initially decreases and reaches a minima. After the minima point, it then increase with increasing K . To get the optimal value of K , you can segregate the training and validation from the initial dataset. Now plot the validation error curve to get the optimal value of K . This value of K should be used for all predictions.

Face Recognition System

A **facial recognition system** is a technology capable of identifying or verifying a person from a digital image or a video frame from a video source. There are multiple methods in which facial recognition systems work, but in general, they work by comparing selected facial features from given image with faces within a database. It is also described as a Biometric Artificial Intelligence based application that can uniquely identify a person by analysing patterns based on the person's facial textures and shape.

While initially a form of computer application, it has seen wider uses in recent times on mobile platforms and in other forms of technology, such as robotics. It is typically used as access control in security systems and can be compared to other biometrics such as fingerprint or eye iris recognition systems. Although the accuracy of facial recognition system as a biometric technology is lower than iris recognition and fingerprint recognition, it is widely adopted due to its contactless and non-invasive process. Recently, it has also become popular as a commercial identification and marketing tool. Other applications include advanced human-computer interaction, video surveillance, automatic indexing of images, and video database, among others.

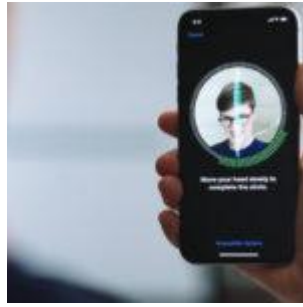


Applications of face recognition System



PREVENT RETAIL CRIME

Face recognition is currently being used to instantly identify when known shoplifters, organized retail criminals or people with a history of fraud enter retail establishments. Photographs of individuals can be matched against large databases of criminals so that loss prevention and retail security professionals can be instantly notified when a shopper enters a store that presents a threat. Face recognition systems are already radically reducing retail crime. According to our data, face recognition reduces external shrink by 34% and, more importantly, reduces violent incidents in retail stores by up to 91%.



UNLOCK PHONES

A variety of phones including the latest iPhone are now using face recognition to unlock phones. This technology is a powerful way to protect personal data and ensure that, if a phone is stolen, sensitive data remains inaccessible by the perpetrator.



SMARTER ADVERTISING

Face recognition has the ability to make advertising more targeted by making educated guesses at people's age and gender. Companies like Tesco are already planning on installing screens at gas stations with face recognition built in. It's only a matter of time before face-recognition becomes an omni-present advertising technology.



FIND MISSING PERSONS

Face recognition can be used to find missing children and victims of human trafficking. As long as missing individuals are added to a database, law enforcement can become alerted as soon as they are recognized by face recognition—be it an airport, retail

store or other public space. In fact, 3000 missing children were discovered in just four days using face recognition in India!



HELP THE BLIND

Listerine has developed a groundbreaking facial recognition app that helps the blind using face recognition. The app recognizes when people are smiling and alerts the blind person with a vibration. This can help them better understand social situations.



PROTECT LAW ENFORCEMENT

Mobile face recognition apps, like the one offered by FaceFirst, are already helping police officers by helping them instantly identify individuals in the field from a safe distance. This can help by giving them contextual data that tells them who they are dealing with and whether they need to proceed with caution. As an example, if a police officer pulls over a wanted murderer at a routine traffic stop, the officer would instantly know that the suspect may be armed and dangerous, and could call for reinforcement.



AID FORENSIC INVESTIGATIONS

Facial recognition can aid forensic investigations by automatically recognizing individuals in security footage or other videos. Face recognition software can also be used to identify dead or unconscious individuals at crime scenes.



IDENTIFY PEOPLE ON SOCIAL MEDIA PLATFORMS

Facebook uses face recognition technology to automatically recognize when Facebook members appear in photos. This makes it easier for people to find photos they are in and can suggest when particular people should be tagged in photos.



DIAGNOSE DISEASES

Face recognition can be used to diagnose diseases that cause detectable changes in appearance. As an example, the National Human Genome Research Institute uses face recognition to detect a rare disease called DiGeorge syndrome, in which there is a portion of the 22nd chromosome missing. Face recognition has helped diagnose the disease in

96% of cases. As algorithms get even more sophisticated, face recognition will become an invaluable diagnostic tool for all sorts of conditions.



RECOGNIZE VIPS AT SPORTING EVENTS

Face recognition can be used to provide fans with a better experience. Face recognition can instantly recognize when season ticketholders attend sporting events. Event venues can offer them swag, let them skip lines and other VIP perks that result in greater season ticketholder retention.



PROTECT SCHOOLS FROM THREATS

Face recognition surveillance systems can instantly identify when expelled students, dangerous parents, drug dealers or other individuals that pose a threat to school safety enter school grounds. By alerting school security guards in real time, face recognition can reduce the risk of violent acts.



TRACK SCHOOL ATTENDANCE

In addition to making schools safer, face recognition has the potential to track students' attendance. Traditionally, attendance sheets can allow students to sign another student, who is ditching class, in. But China is already using face recognition to ensure students aren't skipping class. Tablets are being used to scan students' faces and match their photos against a database to validate their identities.



CASINOS

Face recognition can help casinos recognize the moment that a cheater or advantage gambler enters a casino. In addition, face recognition can recognize members of voluntary exclusion lists, who can cost casinos hefty fines if they're caught gambling.



STOP TOILET PAPER THIEVES

In China, toilet paper theft in public restrooms is a big problem. Luckily face recognition has come to the rescue. China has installed machines in public restrooms that scan people's faces before releasing toilet paper. It won't release more paper to the same person until after 9 minutes have gone by.



FACILITATE SECURE TRANSACTIONS

In China, there is a financial services company called Ant Financial that enables customers to pay for meals by scanning their faces. Customers place orders through a digital menu, and then use face scan as a payment option. After providing their telephone number they can then purchase their meal.



VALIDATE IDENTITY AT ATMS

It seems likely that face scans will eventually replace ATM cards completely since face recognition is such a powerful identity authentication tool. But in the meantime, facerecognition can be used to make sure that individuals using ATMs cards are who they say they are. Face recognition is currently being used at ATMs in Macau to protect peoples' identities.

MAKE AIR TRAVEL MORE CONVENIENT

Airlines have already started using face recognition to help people check bags, check into flights and board planes faster. It seems like we are quickly moving toward a future in which air travel is not only safer than ever before, but also more convenient than any period in history.



TRACK ATTENDANCE AT CHURCH

Churches have started using facial recognition to see which members of their congregation are showing up to church. This can help them identify who to ask for donations and which members to reach out to in order to get them to attend more often.



FIND LOST PETS

Finding Rover is an app that tries to help owners reunite with lost pets. The app uses face recognition (albeit it's the face of an animal in this case) to match photos that pet owners upload to a database of photos of pets in shelters. The app can then instantly alert owners if their pets are found.



RECOGNIZE DRIVERS

More car companies are experimenting with ways to use face recognition. One use of face recognition for automobiles is using a face to replace a key as a means of starting a car. Face

recognition can also be used to change radio stations and seat preferences based on who is driving. Face recognition can even make drivers safer by recognizing and alerting drivers if they are drifting off or not focusing on the road.



CONTROL ACCESS TO SENSITIVE AREAS

Face recognition can work as a means of access control to ensure that only authorized individuals get into facilities like labs, boardrooms, bank vaults, training centers for athletes and other sensitive locations.

CODE FOR DATASET GENERATION

```
#import the libraries
```

```
import numpy as np
```

```
import cv2
```

```
cap = cv2.VideoCapture(0)
```

```
#User information
```

```
label = input("Enter Name: ")
```

```
num = input("Enter Number of Photos to be taken: ") or 10
```

```
num = int(num)
```

```
mugshots = []
```

```
#generation of datasets
```

```
while num:
```

```
    ret, frame = cap.read()
```

```
    if not ret:
```

```
        continue
```

```
# Load haar cascade
```

```
    cascade_classifier = cv2.CascadeClassifier('haarcascade_frontalface_default-  
Copy1.xml')
```

```
# Detect faces
```

```
    faces = cascade_classifier.detectMultiScale(frame, 1.3, 5)
```

```
    faces = sorted(faces, key=lambda e: e[2]*e[3], reverse=True)
```

```
    if not faces:
```

```
        continue
```

```
    faces = [faces[0]]
```

```
#    print(len(faces))
```

```
    for face in faces:
```

```
# Tuple Unpacking
```

```
x,y,w,h = face
```

```
cropped_face = frame[y:y+h, x:x+w]
```

```
# print(cropped_face.shape)
```

```
cropped_face = cv2.resize(cropped_face, (100,100))
```

```
# print(cropped_face.shape)
```

```
mugshots.append(cropped_face)
```

```
num -= 1
```

```
# print(frame.shape)
```

```
cv2.imshow("Feed", frame)
```

```
if cv2.waitKey(1) & 0xFF == ord('q'):
```

```
    break
```

```
mugshots = np.array(mugshots)
```

```
print(mugshots.shape)
```

```
print("Mugshots Taken: ", len(mugshots))
```

```
np.save('face_datasets/' + label, mugshots)
```

```
cap.release()
```

```
cv2.destroyAllWindows()
```

CODE FOR FACE RECOGNITION

```
#import libraries
```

```
import cv2
```

```
import os
```

```
import numpy as np
```

```
from sklearn.neighbors import KNeighborsClassifier
```

```
#using generated datasets
```

```
filenames = os.listdir('face_datasets/')
```

```
labels = [f[:-4] for f in filenames if f.endswith('.npy')]
```

```
print(labels)
```

```
mugshots = []
```

```
for fname in filenames:
```

```
    a = np.load('face_datasets/' + fname)
```

```
    mugshots.append(a)
```

```
print(len(mugshots))
```

```
mugshots = np.concatenate(mugshots, axis=0)
```

```
mugshots = mugshots.reshape((mugshots.shape[0], -1))
```

```
print(mugshots.shape)
```

```
labels = np.repeat(labels, 10)
```

```
labels = labels.reshape(labels.shape[0], -1)
```

```
print(labels.shape)
```

```
dataset = np.hstack((mugshots, labels))
```

```
print(dataset.shape)
```

```
#applying knn algorithm for detecting faces
```

```
knn = KNeighborsClassifier(n_neighbors=5)
```

```
knn.fit(dataset[:, :-1], dataset[:, -1])
```

```
cap = cv2.VideoCapture(0)
```

```
while True:
```

```
    ret, frame = cap.read()
```

```
    if not ret:
```

```
        continue
```

```
    cascade_classifier =
```

```
    cv2.CascadeClassifier('haarcascade_frontalface_default.xml') # Load haar cascade
```

```
# Detect faces
```

```
faces = cascade_classifier.detectMultiScale(frame, 1.3, 5)
```

```
# print(len(faces))
```

```
for face in faces:
```

```
# Tuple Unpacking
```

```
    x,y,w,h = face
```

```
    cropped_face = frame[y:y+h, x:x+w]
```

```
    cropped_face = cv2.resize(cropped_face, (100,100))
```

```
    cropped_face = cropped_face.reshape((1,-1))
```

```
    preds = knn.predict(cropped_face)
```

```
# Drawing a box around the face
```

```
    cv2.rectangle(frame, (x,y), (x+w, y+h), (255,0,0), 2)
```

```
    cv2.putText(frame, preds[0], (x,y), cv2.FONT_HERSHEY_SIMPLEX, 2,  
(0,0,255), 2)
```

```
cv2.imshow("Feed", frame)
```

```
if cv2.waitKey(1) & 0xFF == ord('q'):
```

```
    break
```

```
cap.release()
```

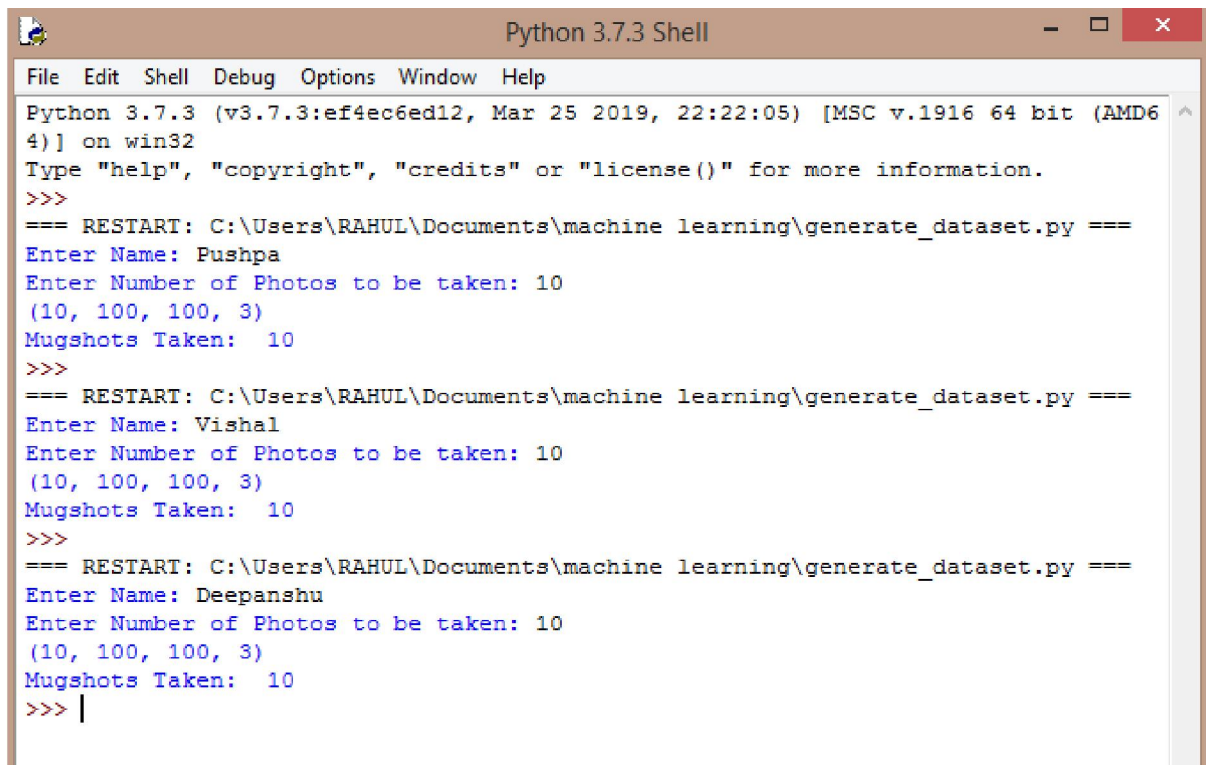

cv2.destroyAllWindows()

DEMONSTRATION OF TECHNOLOGY THROUGH PROJECT

Step 1: Is to generate datasets. It is generated by running generate_dataset.py file. It to enter your name and then how many mugshots you want to take of your face(By default it will take 10 as input if user don't enter anything).Since we

Are using haar-cascade file hence the input can only be face, it means it won't detect any object except face as input.

It looks something like this:-



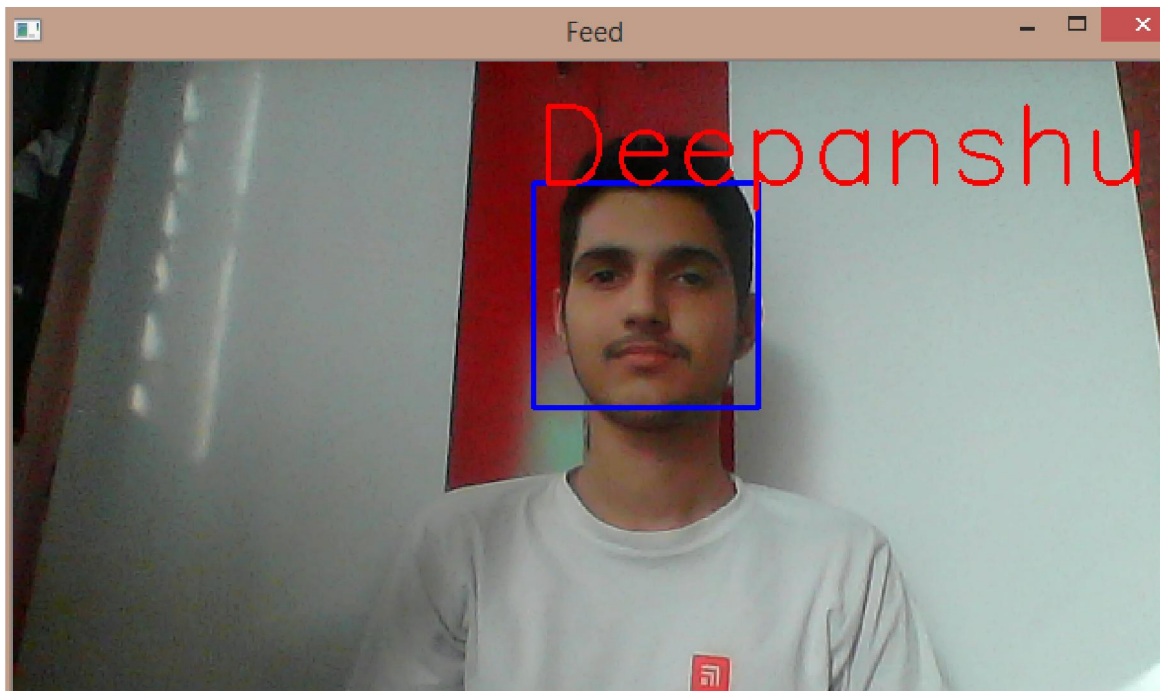
```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: C:\Users\RAHUL\Documents\machine learning\generate_dataset.py ===
Enter Name: Pushpa
Enter Number of Photos to be taken: 10
(10, 100, 100, 3)
Mugshots Taken: 10
>>>
=== RESTART: C:\Users\RAHUL\Documents\machine learning\generate_dataset.py ===
Enter Name: Vishal
Enter Number of Photos to be taken: 10
(10, 100, 100, 3)
Mugshots Taken: 10
>>>
=== RESTART: C:\Users\RAHUL\Documents\machine learning\generate_dataset.py ===
Enter Name: Deepanshu
Enter Number of Photos to be taken: 10
(10, 100, 100, 3)
Mugshots Taken: 10
>>> |
```

Here we can see Pushpa, Vishal and Deepanshu are three persons(they are basically data for our dataset).

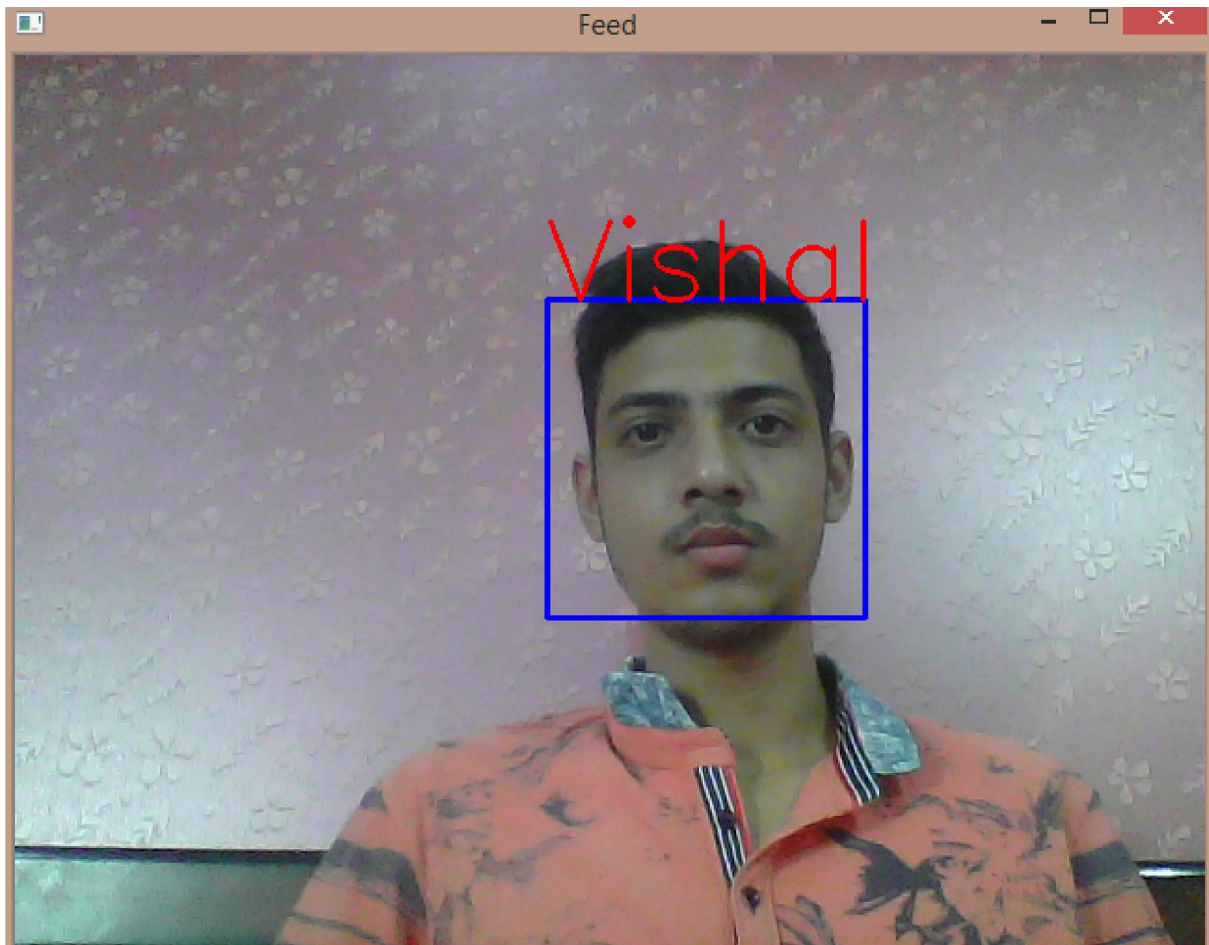
Step 2: The folder in which these datasets are storing is "face_datasets". The photos are storing as "Vikas.npy", "Deepanshu.npy" etc. As shown below:-

| This PC > Documents > machine learning > face_datasets | | | | |
|--|------------------|----------|--------|--|
| Name | Date modified | Type | Size | |
| Deepanshu.npy | 16-09-2019 16:55 | NPY File | 294 KB | |
| Pushpa.npy | 16-09-2019 16:54 | NPY File | 294 KB | |
| Vishal.npy | 16-09-2019 16:55 | NPY File | 294 KB | |

Step 3:- The dataset is generated now its time to check the dataset, so we open face_recognition .np file and after running here are the result:-



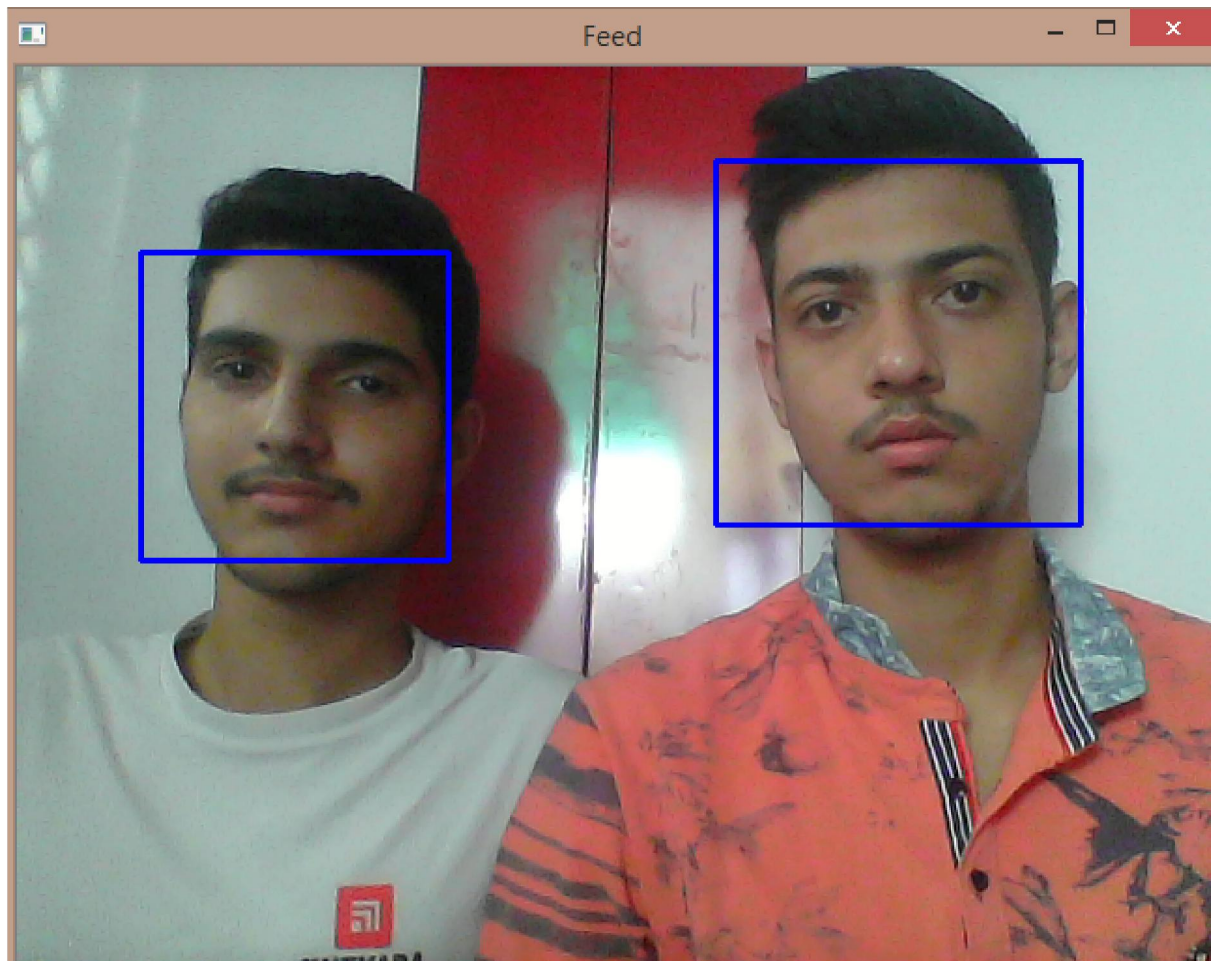
The algorithm runs and identifies that the person is Deepanshu.



And here it shows that the person is Vishal.

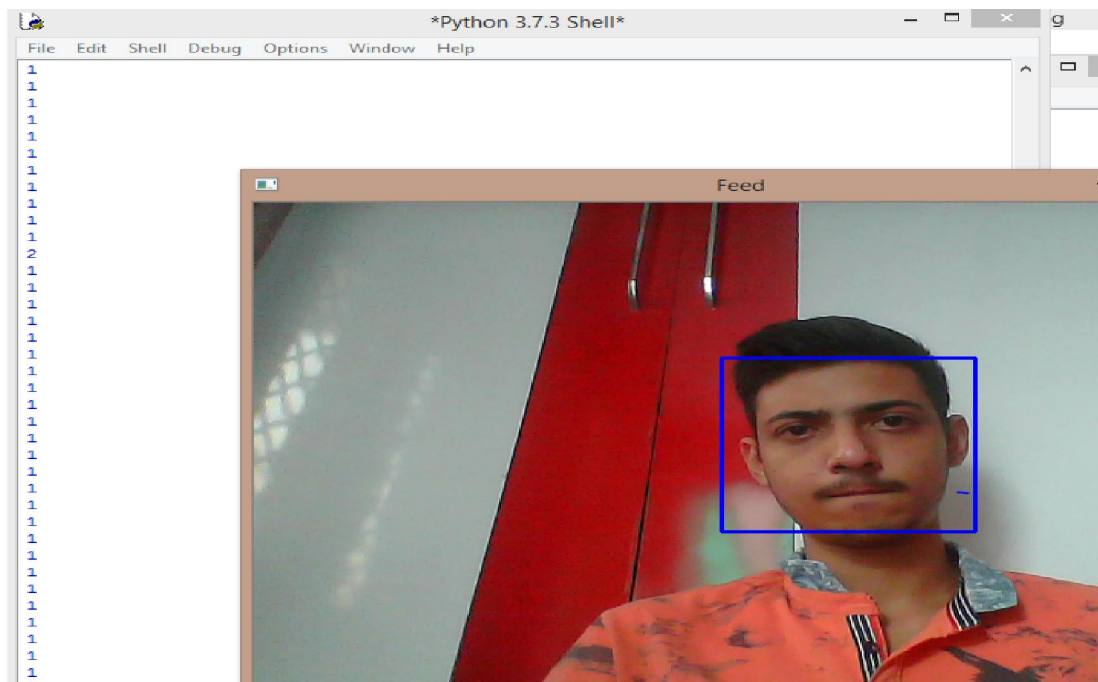
Similarly we have face detection program which identifies a person face irrespective by its name because it's data is not submitted. There is difference between face detection and face recognition. In face detection the algorithm determines whether a person is present or not whereas in face recognition the person face is match with the data present in the database.

Here is the example of face detection:-



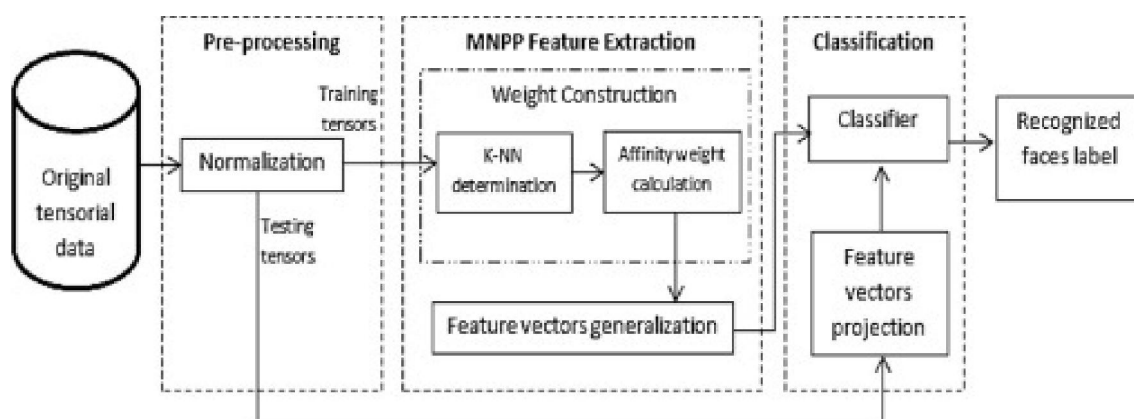
Here it identifies faces, not recognise.

Basically what is happening in the background is whenever the camera is seeing a face it is changing its count to 1, if two person are present then it changes its count to 2 and so on. But if no person is detected it shows 0 on screen.



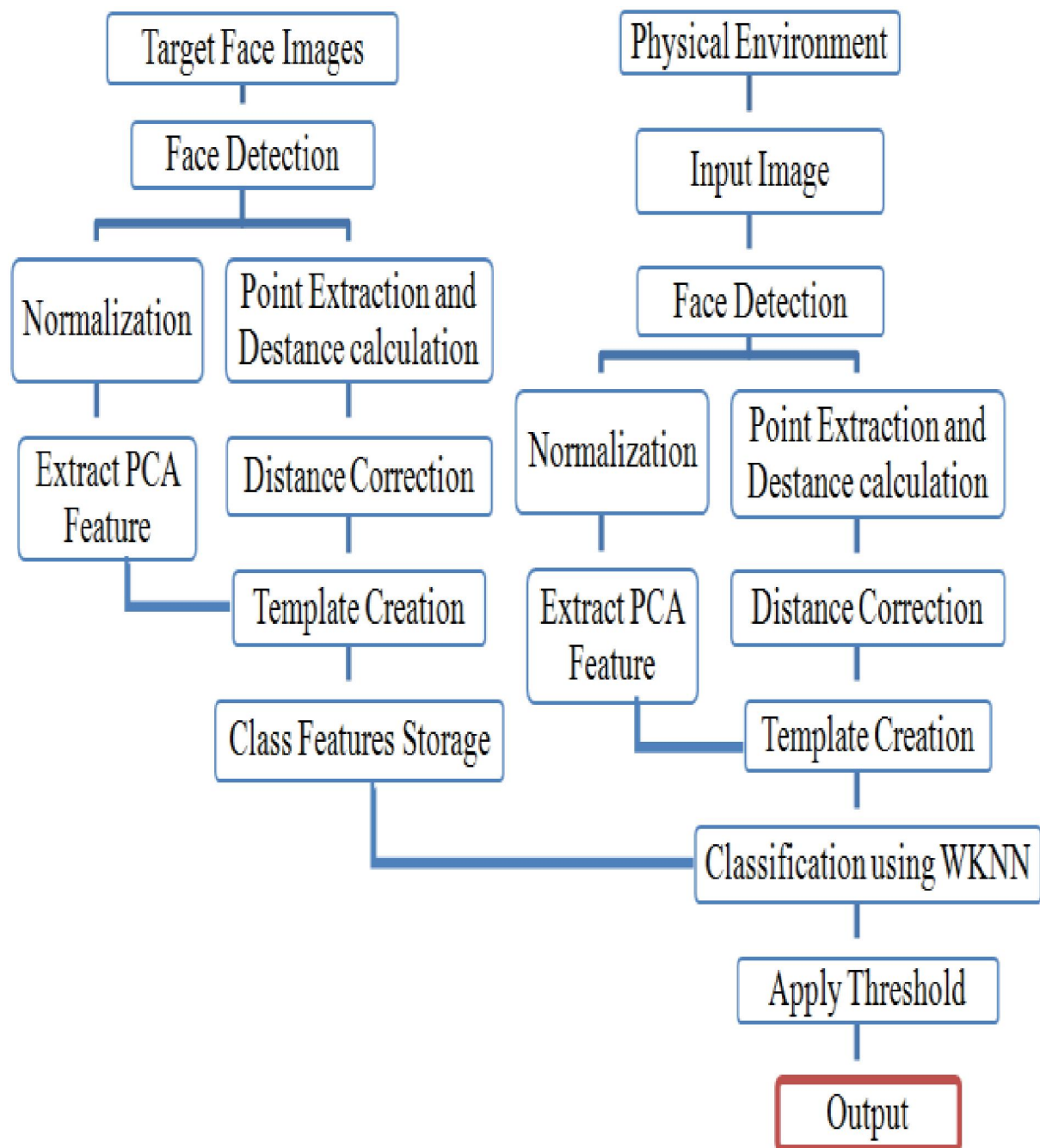
Thats all..... how this projects works.

SEQUENCE DIAGRAM



It is a live project i.e it uses web cam for detection of a person or its face, first the dataset is generated and stored into face_datasets/ file. So our preprocessing task get completed and our dataset is ready for testing purpose. Now, when we run face recognition code the mugshots are taken and the face of the person is compared with the faces present in the dataset. The algorithm used for detecting the face here

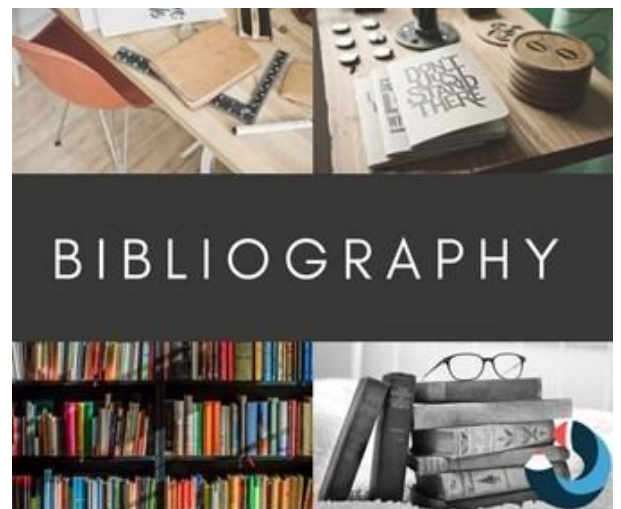
is KNN(k nearest neighbour), it classify the image and make a square around the input test face which implies that the face is detected and if the face is present in the dataset then name of the person will also be displayed.



CONCLUSION

It was a wonderful learning experience for me while working on this project. This project took me through the various phases of project development and gave me a real insight into the world of software engineering. The joy of working and the thrill involved while tackling the various problems and challenges gave me a feel of developers industry. It was due to this project I came to know how professional software is designed. The project works fine but there is a lot of scope of improvement. I think it can further be improved with the help of Neural Networks. It can work more properly on huge number of datasets and more accurately using neural networks. It only determine face not any other objects, the face detection program can detect multiple faces at a time. Whenever a face is detected my machine it shows output as 1 or other numeric counts depends upon number of faces as inputs, and when no face is present then it shows 0 as output. Really nice experience with learning machine learning, it is one of the most popular thing which everyone is trying to learn, it is changing our future very fast and in effective manner making human lives easier and safe.

BIBLIOGRAPHY



Books:

- Hastie, Friedman, and Tibshirani, The Elements of Statistical Learning, 2001
- Bishop, Pattern Recognition and Machine Learning, 2006
- Ripley, Pattern Recognition and Neural Networks, 1996
- Duda, Hart, and Stork, Pattern Classification, 2nd Ed., 2002
- Scholkopf and Smola, Learning with Kernels, 2002
- Computational Statistics (online book)
- MIT Papers

Other machine learning courses:

- Andrew Ng
- Max Welling

Data repositories collected from

- Keras
- Sklearn
- Google tensorflow

Other Sources:

- Google
- Wikipedia
- Keras
- Youtube
- Nptel
- Udacity Programs
- GeeksforGeeks

APPENDICES

You can also go through these report paper, documentaries and books for outside the box knowledge

- Allix, K, Bissyandé, T. F., Klein, J., Traon, Y. L. (2014). Machine Learning-Based Malware Detection for Android Applications: History Matters!. Retrieved on December 20, 2017 from

http://publications.uni.lu/bitstream/10993/17251/1/history_matters.pdf

- Greene, T. (2017). Google brings on-device machine learning to mobile with TensorFlow Lite. Retrieved December 20, 2017 from

<https://thenextweb.com/artificial-intelligence/2017/11/15/google-brings-on-device-machine-learning-to-mobile-with-tensorflow-lite/>

- Jennings, S. (2017). Expert Interview: Michael Schmidt of Nutonian on the Future of Machine Intelligence. Syncsort Trillium Software. Retrieved December 22, 2017 from <http://blog.syncsort.com/2017/02/big-data/expert-interview-schmidt-nutonian-machine-intelligence/>

- Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science* (New York, N.Y.), 349(6245), 255.
doi:10.1126/science.aaa8415

- Marceau, F. (2017). How to use a machine learning model on iOS 11 using Core ML. Towards Data Science. Retrieved December 20, 2017 from

<https://towardsdatascience.com/how-to-use-a-machine-learning-model-on-ios-11-using-core-ml-24a9d8654df6>

- Matyunina, Julia. (2017). The Ultimate Guide to Machine Learning in Mobile Apps. Codetiburion. Retrieved December 19, 2017 from <https://codetiburion.com/ultimate-guide-machine-learning-in-mobile-apps/>
- Mullainathan, S., & Spiess, J. (2017). Machine Learning: An Applied Econometric Approach. Journal of Economic Perspectives, 31(2), 87-106. doi:10.1257/jep.31.2.87
- Ravindra, S. (2017). How Convolutional Neural Networks Accomplish Image Recognition? KDNuggets. Retrieved January 5, 2018 from <https://www.kdnuggets.com/2017/08/convolutional-neural-networks-image-recognition.html>
- Schatsky, D. (2016). Machine Learning is Going Mobile. Deloitte University Press. Retrieved December 24, 2017 from https://www2.deloitte.com/content/dam/insights/us/articles/machine-learning-mobile-applications/DUP_2830_Machine-learning_vFINAL.pdf
- Simonite, T. (2017). Artificial Intelligence Seeks an Ethical Conscience. Wired. Retrieved January 2, 2018 from <https://www.wired.com/story/artificial-intelligence-seeks-an-ethical-conscience/>