

**A  
PROJECT REPORT  
ON  
Gender Recognition and Age Detection using Human  
Facial Features**

*Submitted by*

**Jayraj Malamdi (19IT424)  
Bhavya Dave (19IT433)  
Shrey Bhatt (19IT463)  
Dhruv Shah (19IT425)**

**For Partial Fulfillment of the Requirements for Bachelor of Technology in  
Information Technology**

**Guided by**

**Dr. Zankhana Shah**

**December, 2022**



**Information Technology Department  
Birla Vishvakarma Mahavidyalaya Engineering College  
(An Autonomous Institution)  
Vallabh Vidyanagar – 388120  
Gujarat, INDIA**



**Birla Vishvakarma Mahavidyalaya Engineering College**

**(An Autonomous Institution)**

**Information Technology Department**

**AY: 2022-23, Semester I**

## **CERTIFICATE**

This is to certify that the project work entitled **“Gender Recognition And Age Detection Using Human Facial Features”** has been successfully carried out by **Jayraj Malamdi (19IT424), Bhavya Dave (19IT433), Shrey Bhatt (19IT463), Dhruv Shah (19IT425)** for the subject **Project I (4IT31)** during the academic year 2022-23, Semester-I for partial fulfilment of Bachelor of Technology in Information Technology. The work carried out during the semester is satisfactory.

Dr. Zankhana Shah  
IT Department  
BVM

Dr. Keyur Brahmhatt  
Head, IT Department  
BVM

# ACKNOWLEDGEMENT

In this page I would like to say thank you to a number of people who supported us while preparing this Project work .

First of all, we are indebted to the GOD ALMIGHTY for giving us an opportunity to excel in our efforts to complete this project on time.

We are extremely grateful to **Dr. Indrajit Patel**, Principal, Birla Vishvakarma Mahavidyalaya Engineering College and **Dr. Keyur Brahmhatt**, Head of the Information Technology Department, for providing all the required resources for the successful completion of our project.

Our heartfelt gratitude to our project guide **Dr. Zankhana Shah** , Assistant Professor in Information Technology, for her valuable suggestions and guidance in the preparation of the progress report and implementation of project.

We express our thanks to **project co-ordinator Dr. Zankhana Shah, all staff members and friends**, for all the help and co-ordination extended in bringing out this project successfully in time.

We will be failing in duty if we do not acknowledge with grateful thanks to the authors of the references and other literatures referred to in this project.

Last but not the least, we are very much thankful to our parents who guided us in every step which we too.

# **ABSTRACT**

Information extraction from the human face is one of the active research areas in recent years. Numerous studies have been conducted on the recognition of the most common face variant, including identification, age and gender. This study suggests a convolutional neural network and support vector machine- based method for automatically identifying a person's age and gender from their face. Pre-processing, Face Normalization, Feature Extraction, and Classification are the four stages of this method. Additionally, our system includes a number of deep convolutional neural networks that have been trained to identify faces in live feed and estimate their age and gender. Using attention mechanism enables our model to focus on the important and informative parts of the face, which can help it to make a more accurate prediction. Our model is trained on a popular face age and gender dataset, and achieved promising results.

## LIST OF FIGURES

Figure 1: k-spilt	10
Figure 2: Combined Age and Gender Dataset	11
Figure 3: Age Dataset for Male	11
Figure 4: Age Dataset for Female	11
Figure 5 : Raw Images in Adience Benchmark Dataset	12
Figure 6: MTCNN process	13
Figure 7: Class balance	14
Figure 8: Curse of Dimensionality	16
Figure 9: Information preserved by F2 >> information preserved by F1	17
Figure 10: Information preserved by F2 = information preserved by F1	17
Figure 11: After applying PCA	18
Figure 12: Train & Test split	20
Figure 13: SVM	21
Figure 14: CNN Architecture 1	22
Figure 15: Flattening of a 3x3 image matrix into a 9x1 vector	23
Figure 16: 4x4x3 RGB Image	24
Figure 17: The first block is encircled in black	25
Figure 18: The second block is encircled in black	26
Figure 19: CNN architecture 2	30
Figure 20: Training Loss vs Validation Loss Plot	33
Figure 21: Training Accuracy vs Validation Accuracy for Age	33
Figure 22: Training Accuracy vs Validation Accuracy for Gender	34

Figure 23: Confusion Matrix of SVM age	34
Figure 24: Confusion Matrix of SVM gender	35
Figure 25: Confusion Matrix of CNN age	35
Figure 26: Confusion Matrix of CNN gender	35
Figure 27: Use Case Diagram	36
Figure 28: ER Diagram	37
Figure 29: Sequence Diagram	37
Figure 30: Data Flow Diagram (Level 0)	38
Figure 31: Data Flow Diagram (Level 1)	38
Figure 32: User Interface of Home Page	41
Figure 33: Output of CNN Model	42
Figure 34: Output of SVM Model	42

## LIST OF TABLES

1. Timeline Chart .....	9
2. Dataset Analysis.....	10

## **List Of Abbreviations**

1. CNN : Convolution Neural Networks
2. SVM : Support Vector Machine.
3. FD : Face Detection
4. MTCNN : Multi-Task Convolution Neural Networks



# INDEX

<b>1. INTRODUCTION.....</b>	<b>1</b>
1.1. Brief Overview of the work.....	1
1.2. Objective.....	1
1.3. Scope.....	1
1.4. Project Modules.....	2
1.5. Project Requirements.....	2
1.5.1 Hardware Requirements.....	2
1.5.2 Software Requirements.....	2
<b>2. LITERATURE REVIEW.....</b>	<b>5</b>
<b>3. SYSTEM ANALYSIS &amp; DESIGN.....</b>	<b>8</b>
3.1. Project Feasibility Study.....	8
3.2. Project Timeline.....	9
3.3. Details Modules Description.....	9
3.3.1 Data Acquisition.....	9
3.3.2 Face Detection.....	12
3.3.3 Data Preprocessing.....	14
a) Data Augmentation.....	14
b) Data Normalisation.....	15
c) Dimensionality Reduction.....	15

3.3.4 Model Building.....	20
a) SVM.....	21
b) CNN.....	22
3.3.5 Model Training.....	30
3.3.6 Model Evaluation.....	33
3.4 PROJECT SRS.....	36
3.4.1 Use Case Diagram.....	36
3.4.2 ER Diagram.....	37
3.4.3 Sequence Diagram.....	37
3.4.4 Data Flow Diagram.....	38
<b>4. IMPLEMENTATION &amp; TESTING.....</b>	<b>39</b>
4.1 User Interface and Snapshot.....	39
4.2 Testing using Use Case.....	42
<b>5. CONCLUSION &amp; FUTURE WORK.....</b>	<b>43</b>
<b>6. REFERENCES.....</b>	<b>44</b>



# Chapter 1 : INTRODUCTION

## 1.1 Brief Overview of the work

Age and gender information are crucial for many real-world applications, including behavior research, online advertising, social understanding, identity verification, video surveillance, human-computer interface and many more. Despite their wide range of uses, automatically determining age and gender from facial photos is a challenge. This is largely because there are many intra-class variances in people's facial images. The project's goal is to use a web application made with flask to estimate a person's age and gender from a live feed.

## 1.2 Objective

- In this Project, we have done face detection and we have used Keras and TensorFlow for age and gender prediction.
- To build a gender and age detector that can approximately guess the gender and age of the person(face) in a picture using Deep Learning on Adience Dataset.
- The predicted gender may be one of 'Male' and 'Female', and the predicted age may be one of the following ranges-

(0– 2), (4 – 6), (8 – 12), (15 – 20), (25 – 32), (38 – 43), (48 – 53), (60 –100).

## 1.3 Scope

- This field of study has a huge amount of underlying potential. There has been an ever-growing interest in automatic age and gender prediction because of the huge potential it has in various fields of computer science such as HCI (Human Computer Interaction).
- This study's focus is restricted to a few different variables. The first is that the subject should have a frontal face and be facing the camera, as opposed to having a sideways gaze. The second is that there should be adequate lighting in the space where the application is made. The person's face should not be concealed, as this would make it harder to retrieve information from the face.

## **1.4 Project Modules**

- 1.4.1 Data Acquisition
- 1.4.2 Face Detection
- 1.4.3 Data PreProcessing
- 1.4.4 Model Building
- 1.4.5 Model Training
- 1.4.6 Model Evaluation

## **1.5 PROJECT REQUIREMENTS**

### **1.5.1 Hardware Requirements:**

- For development – Jupyter Notebook.
- For user – Computer is required to upload an image.
- As the dataset is small, so we can load it entirely from hard drive to memory for faster processing. We do not need any dataset loader for this problem.

### **1.5.2 Software Requirements:**

- We have created an environment named “TF” with python version 3.7 or higher, in which we have installed these packages:
  - Tensorflow 2.10.0
    - Tensorflow 2.10.0 is the latest version of TensorFlow.
    - It is a free and open-source software library, used for training and inference of deep neural networks.
  - OpenCV-python
    - OpenCV is a great tool for image processing and performing computer vision tasks.
    - It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more.

- Pandas
  - Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive.
- tqdm pillow scikit-learn flask
  - tqdm is a library in python which is used for creating Progress Meters or Progress Bars.
  - Pillow Imaging Library is a free and open-source library that contains all the basic image processing functionality.
  - Scikit-learn (Sklearn) provides a selection of efficient tools for machine learning and statistical modeling in python.
- Matplotlib seaborn
  - Seaborn is built on the roof of Matplotlib and is considered as a superset of the Matplotlib library. It helps in visualizing univariate and bivariate data.
  - Together it offers sane choices for plot style and color defaults, defines simple high-level functions for common statistical plot types, and integrates with the functionality provided by Pandas DataFrames.
- Torch
  - Torch is an open-source ML library used for creating deep neural networks and is written in the Lua scripting language. It's one of the preferred platforms for deep learning research.
  - The framework is built to speed up the process between research prototyping and deployment. It also exports learning models to the Open Neural Network Exchange (ONNX) standard format and also Offers a user-friendly interface.

- Torchvision
  - Torchvision is a library for Computer Vision that goes hand in hand with PyTorch. It has utilities for efficient Image and Video transformations, some commonly used pre-trained models, and some datasets
  
- Flask
  - Flask is a micro web framework, used for developing web applications.
  - It serves as a foundation for software developers, allowing them to create a variety of applications for certain platforms. It is a set of functions and predefined classes used to connect with the system software and handle inputs and outputs.

## **Chapter 2 : LITERATURE REVIEW**

### **1. Age And Gender Detection (Published by International Journal of Scientific Research in Engineering and Management (IJSREM))**

- It is written by Dr.C.K.gomathy, A.Lokesh ,Ch.Harshavardhan Reddy ,A.Sai Kiran in the Year 2021 .
- It uses SVM, naive bayes, decision tree as its techniques to build the model.
- Advantages - The summary of the model is decent and better than many already existing models.
- Disadvantages - In this, points have been explained in depth. In this they demonstrate similar gains with a simple network architecture designed by considering the rather limited availability of age and gender labels in existing cases.

### **2. Human gender, classification using machine learning (Published by International Journal of Engineering Research and Technology (IJERT))**

- It is written by Miss Vaishnavi Mali, Dr. Babasaheb G. Patil in the Year 2020.
- It uses LDA, CNN, SVM as its techniques to build the model.
- Advantages - Recognition of age and gender by computers give more accuracy as compared with human, this paper provides a detailed review of a system for human gender classification.
- Disadvantages - It is less accurate in some algorithms. This method is unachievable in case of person wearing glasses



**3. Human Age and Gender Estimation using Facial Image Processing (Published by IEEE)**

- It is written by Syed Taskeen Rahman, Asiful Arefeen, Shashoto Sharif Mridul, Asir Intisar Khan, Samia Subrina in the year 2020.
- It uses Edge detection prediction, naive Bayes classification, BUET facial database as its technique to build the model.
- Advantages - Image processing-based method involving comparison of some feature extracted from the post processed facial images of people of various age ranges.
- Disadvantages - Less accuracy, Only 76.3% accuracy in predicting the age

**4. Age Prediction using Image Dataset using Machine Learning (Published by International Journal of Innovative Technology and Exploring Engineering (IJITEE)).**

- It is written by Ishita Verma, Urvi Marhatta, Sachin Sharma, Vijay Kumar in the year 2019.
- It uses CNN, Keras, Scipy, tensorflow as its technique to build the model.
- Advantages - CNN can be used to improved age and gender classification.
- Disadvantages - RNN is not applicable for this paper because it takes text or speech as an input whereas in this it takes only image.

**5. Estimation Of Age Group-based on Facial Features (Published by International Journal of Engineering Research & Technology (IJERT)).**

- It is written by Rahul Kumar, Akhilesh Kumar Srivastava, Deepak Kumar Agarwal in the year 2018 .
- It uses Viola Jones Algorithm, K-means Clustering as its technique to build the model.
- Advantages - Uses local binary patterns and geometric patterns for achieving the best execution .From geometric elements, the cross proportion is determined.
- Disadvantages - Not used logistic regression, Naive Bayes Algorithm.

**6. Geometric and appearance feature analysis for facial expression recognition  
(Published by International Journal of Advanced Engineering Technology).**

- It is written by Sonu Dhall, Poonam Sethi in the year 2014.
- It uses Cohn- kanade database, SVM classifier.
- Advantages - It uses 3 algorithms: decision tree, random forest, naive Bayes.
- Disadvantages - Gender classification is a binary problem in which data assigned is either male or female.

**7. Age and Gender Classification using Convolutional Neural Networks (Published by  
The Open University of Israel)**

- It is written by Gil Levi ,Tal Hassner in the year 2015.
- It uses Deep CNN, Adience dataset as its technique to build the model.
- Advantages - Investigated LBP-TOP features for low-resolution facial expression recognition.
- Disadvantages - If the image contains some degree of tilt or rotation then CNN usually have difficulty in identifying image.

## **Chapter 3 : System Analysis & Design**

### **3.1 Project Feasibility Study**

The following feasibilities are considered in our project, in order to ensure that the project does not have any major obstructions.

#### **3.1.1 Technical feasibility:**

- Our project is technical feasible because the most recent versions of Jupyter Notebook, Python, Pandas, Matplotlib, Seaborn, NumPy, Flask, and Microsoft Word have been used to construct this system.
- Technically, all of the aforementioned Applications are possible and readily available in market.

#### **3.1.2 Operational Feasibility:**

- Our project is operationally feasible because the service is adaptable and scalable.
- Time and human interaction are all utilized to the utmost extent.
- It offers dependable services which is highly precise system.

#### **3.1.3 Implementation Feasibility:**

- Our project is been designed with a user-friendly interface so that anybody may access it.
- The webcam records the subject's face information, and the output is shown, on a click of a button.

## 3.2 Project Timeline Chart

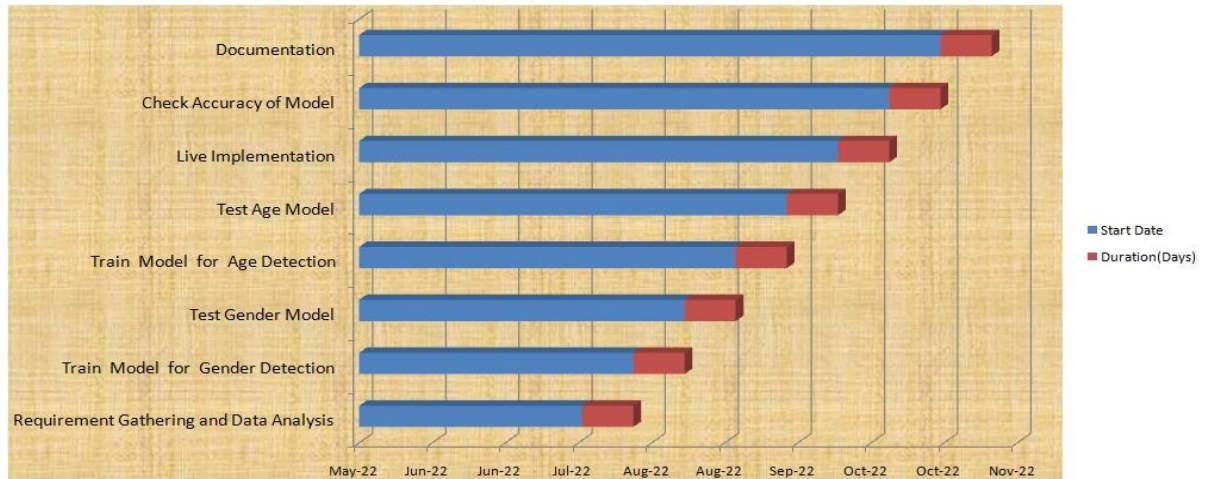


Table 1: Timeline Chart

## 3.3 Detailed Modules Description

### 3.3.1 Data Acquisition

- The Adience benchmark dataset is selected as our dataset for detecting the age and gender in images.
- The entire Adience collection includes 26,580 256×256 color facial images of 2,284 subjects, with eight age group classes:  
(0 – 2), (4 – 6), (8 – 13), (15 – 20), (25 – 32), (38 – 43), (48 – 53), (60 – 100)
- Adience dataset comes from images that people automatically upload to network albums from smart phones. These images are not artificially filtered before uploading, and they are completely unconstrained as they were taken under different variations. We use the in-plane face aligned method to align faces.
- Testing for age classification is performed using a standard five-fold, subject exclusive cross-validation protocol, and the accuracy of five folds are averaged to be the final age group classification result.

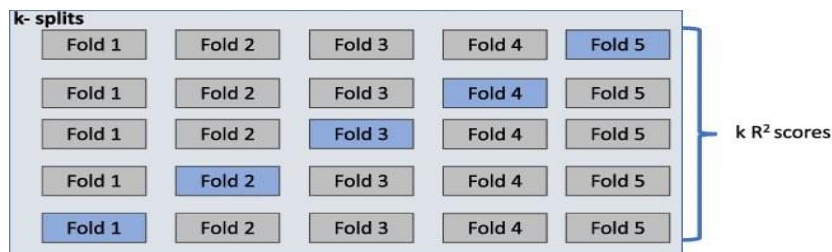


Figure 1: k-spilt

- Breakdown of the Adience benchmark into different age and Gender classes.

Age	Male	Female	Total
4—6	906	1234	2140
8—12	934	1190	2124
15-20	734	908	1642
25-32	2308	2696	5004
38-43	1294	999	2293
48-53	392	428	820
60-100	376	496	872
38-48	2	4	6
35	123	170	328
3	8	10	21
55	33	43	131
58	2	3	63
22	60	89	171
13	66	102	181
45	55	33	133
36	24	12	72
None	455	293	748

Table 2: Dataset Analysis

- **Bar Graph:**

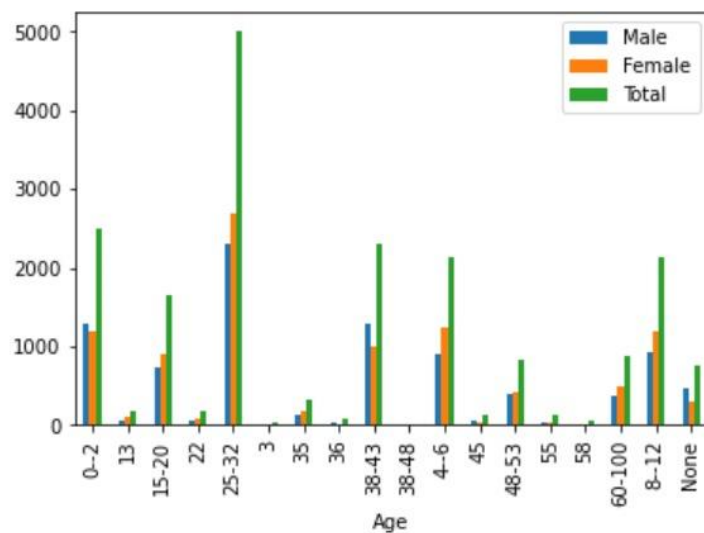


Figure 2: Combined Age and Gender Dataset

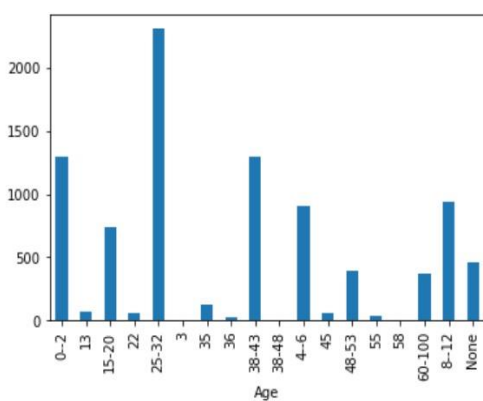


Figure 3: Age Dataset for Male

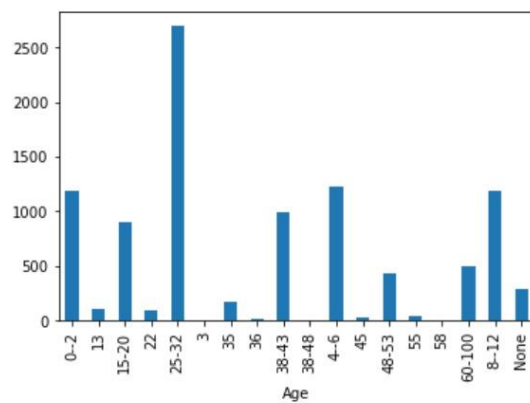


Figure 4: Age Dataset for Female

- Sample Images from Adience Benchmark Dataset:



Figure 5 : Raw Images in Adience Benchmark Dataset

### 3.3.2 Face Detection

- First created a function named “cropPhoto” which crops the photo.
- For face detection we have used, MTCNN(multi-Task Cascaded CNN). It is capable of recognizing other facial features such as eyes & mouth, called **Landmark detection**.
- The network uses a cascade structure with three networks:
  - First the image is rescaled to a range of different sizes (called an image pyramid), then the first model (Proposal Network or P-Net) proposes candidate facial regions, the second model (Refine Network or R-Net) filters the bounding boxes, and the third model (Output Network or O-Net) proposes facial landmarks.
- The model is called a multi-task network because each of the three models in the cascade (P-Net, R-Net and O-Net) are trained on three tasks, e.g., make three types of predictions they are: face classification, bounding box regression, and facial landmark localization.

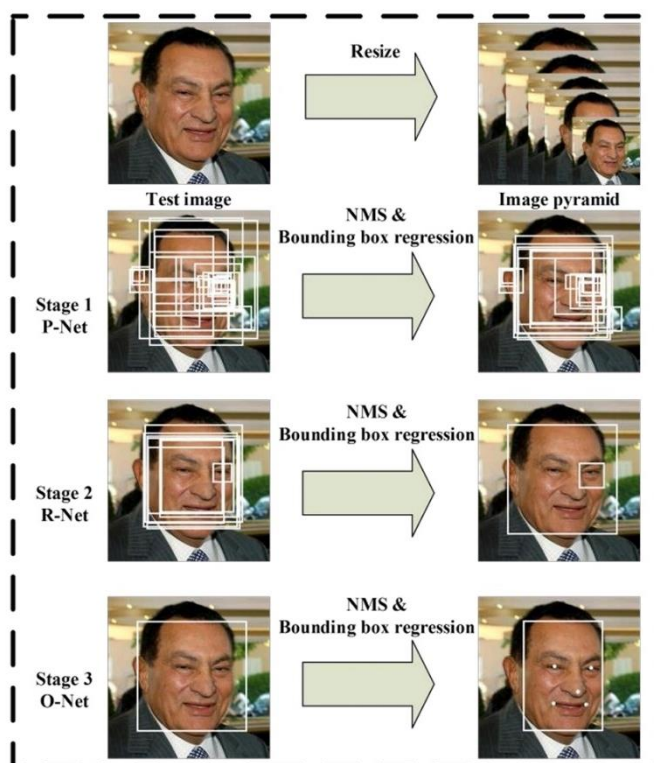


Figure 6: MTCNN process

- The above image is taken from the paper[8], which provides a helpful summary of the three stages from top-to-bottom and the output of each stage left-to-right.
- The three models are not connected directly; instead, outputs of the previous stage are fed as input to the next stage. This allows additional processing to be performed between stages; for example, non-maximum suppression (NMS) is used to filter the candidate bounding boxes proposed by the first-stage P-Net prior to providing them to the second stage R-Net model.



### 3.3.3 Data Preprocessing

#### a) Data Augmentation

- Data augmentation is a process of artificially increasing the amount of data by generating new data points from existing data. This includes adding minor alterations to data or using machine learning models to generate new data points in the latent space of original data to amplify the dataset. Data augmentation methods are widely used in practically every cutting-edge deep learning application such as object detection, image classification, image recognition, natural language understanding, semantic segmentation, and more. Augmented data is improving the performance and results of deep learning models by generating new and diverse instances for training datasets.
- We will be using data augmentation to bring class balance to the dataset.

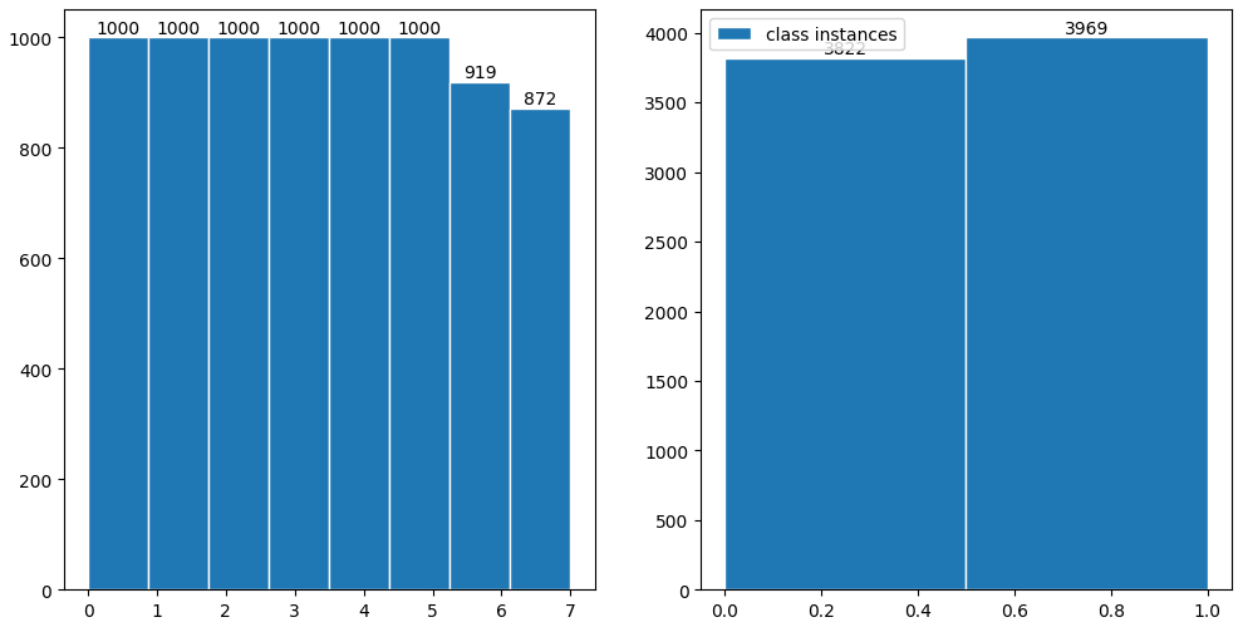


Figure 7: Class balance

## b) Data Normalization

- Also referred to as data re-scaling, it is the process of projecting image data pixels (intensity) to a predefined range e.g. [0, 1]. This is commonly used on different data formats, and you want to normalize all of them to apply the same algorithms over them.
- Normalization is usually applied to convert an image's pixel values to a typical or more familiar sense. Its benefits include:
  - Fairness across all images - For example, scaling all images to an equal range allows all images to contribute equally to the total loss rather than when other images have high and low pixels ranges give strong and weak loss, respectively.
  - Provides a standard learning rate - Since high pixel images require a low learning rate and low pixel images high learning rate, re-scaling helps provide a standard learning rate for all images.
- Here, we are normalizing our RGB images which are of data type uint8 and are in the range of [0, 255]. The new range will be [0, 1].

## c) Dimensionality Reduction

- Scikit-learn Classifiers accepts the training features to be in 2D arrays, which means we can't feed RGB images directly into any scikit-learn classifier. To tackle this situation, we need to reshape or flatten the 3d RGB images into 1D array. But by simply flattening the an RGB image of shape 256x256x3, it will become an 1D array of shape 1x196608 which has a lot of features in it.
- Due to which data has become very large and complex in which a certain outcome or a decision is dependent on not just a single factor (predictor) but on multiple factors that complicate the decision-making process. This is called the **CURSE OF DIMENSIONALITY**.
- Due to higher dimension, the data becomes sparse due to which ML algorithm fails also. Due to this complexity, the training time is increased exceedingly and the model will perform poorly.

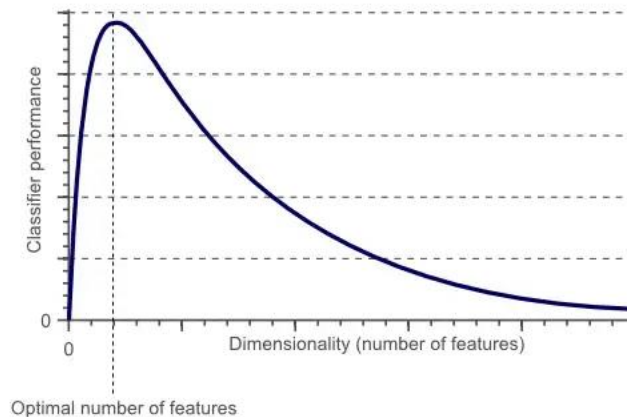


Figure 8: Curse of Dimensionality

- Hence it becomes very critical to reduce the number of factors or features to a few important ones to arrive at the right decision. This process is called **Dimensionality Reduction**.
- The problem of high dimensionality is dealt in two ways:
  - 1. **Feature selection** — is carefully selecting the important features by filtering out the irrelevant features.
  - 2. **Feature extraction** — is creating new and more relevant features from the original features. Principal Component Analysis (PCA) is one of the key techniques of feature extraction.

### **PCA (Principal Component Analysis) :**

- Here to tackle this problem, we performed dimensionality reduction technique by applying PCA (Principal Component Analysis) in SVM model.
- Principal component analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set.

- PCA can be used when we want to:
  1. Reduce the number of features but cannot identify the unimportant ones that can be ignored,
  2. Ensure that the features of the data are independent of one another even if the features become less interpretable.
- Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to explore and visualize and make analysing data much easier and faster for machine learning algorithms without extraneous variables to process.
- Geometric Intuition:

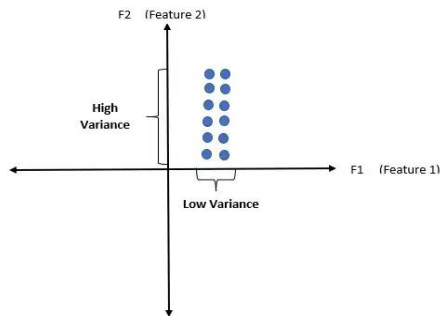


Figure 9: Information preserved by F2 >> information preserved by F1

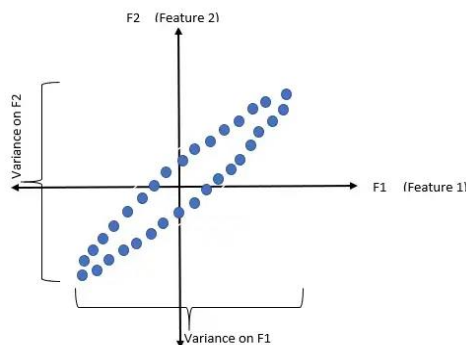


Figure 10: Information preserved by F2 = information preserved by F1

- Here, PCA comes into the picture. So, PCA performs linear orthogonal transformation on the data to find features  $F1'$  and  $F2'$  such that the variance on  $F1' \gg$  variance on  $F2'$ .
- These new sets of features are called principal components. Here,  $F1'$  is the first principal component which gives the direction of maximum variance, and  $F2'$  is the second principal component which gives the second direction with most variance.
- $F1'$  and  $F2'$  are unit vectors and they are perpendicular to each other.
- Since  $F1'$  and  $F2'$  are our new set of features we can safely drop  $F2'$  as  $F1'$  preserves maximum information.

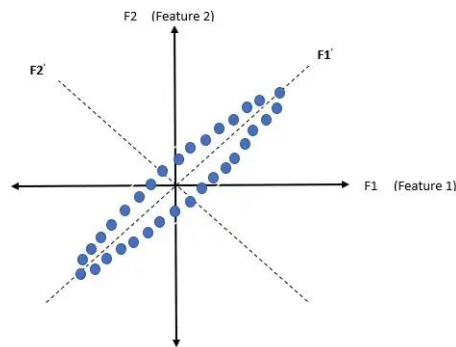


Figure 11: After applying PCA

- Here in our project, we have taken the first 500 components which covers up to 90% information when summed up the individual component's variance. This means we have successfully reduced 196608 features to just 500 features by losing only 10% of the information from the dataset.

After PCA is applied on Gender dataset for SVM model :

```
In [11]: # PCA can be easily done by using scikit-Learn PCA package
from sklearn.decomposition import PCA
import numpy as np
import pickle

# First reshape the batch of RGB images which is a 4D array into a 2D array.
print("Shape of the original dataset:", X_normalized.shape)
X_resaped = X_normalized.reshape((len(X_normalized), -1))
print("Shape of the dataset when flattened:", X_resaped.shape)

pca = PCA(n_components=50)
pca.fit(X_resaped)

print("Sum of explained variance ratio for each Principle Component:", sum(pca.explained_variance_ratio_))

# as we have PCA object, now transform the original dataset into a new dataset with 100 features
X_PCA = pca.transform(X_resaped)

print("Shape of the dataset after applying PCA:", X_PCA.shape)

# Save PCA object which has been trained on the training set
pca_name = 'data/Dataset_Gender_PCA.pkl'
pickle.dump(pca, open(pca_name, "wb"))

Shape of the original dataset: (19338, 128, 128, 3)
Shape of the dataset when flattened: (19338, 49152)
Sum of explained variance ratio for each Principle Component: 0.800620214547962
Shape of the dataset after applying PCA: (19338, 50)
```

After PCA is applied on Age dataset for SVM model :

```
In [135]: # PCA can be easily done by using scikit-Learn PCA package
from sklearn.decomposition import PCA
import numpy as np
import pickle

# First reshape the batch of RGB images which is a 4D array into a 2D array.
print("Shape of the original dataset:", X_normalized.shape)
X_resaped = X_normalized.reshape((len(X_normalized), -1))
print("Shape of the dataset when flattened:", X_resaped.shape)

pca = PCA(n_components=100)
pca.fit(X_resaped)

print("Sum of explained variance ratio for each Principle Component:", sum(pca.explained_variance_ratio_))

# as we have PCA object, now transform the original dataset into a new dataset with 100 features
X_PCA = pca.transform(X_resaped)

print("Shape of the dataset after applying PCA:", X_PCA.shape)

# Save PCA object which has been trained on the training set
pca_name = 'data/Dataset_PCA.pkl'
pickle.dump(pca, open(pca_name, "wb"))

Shape of the original dataset: (15731, 128, 128, 3)
Shape of the dataset when flattened: (15731, 49152)
Sum of explained variance ratio for each Principle Component: 0.8617803838569671
Shape of the dataset after applying PCA: (15731, 100)
```

### 3.3.4 Model Building

- After the preprocessing is done, the dataset is split for train and test purposes. Test size is 20% of the train size.

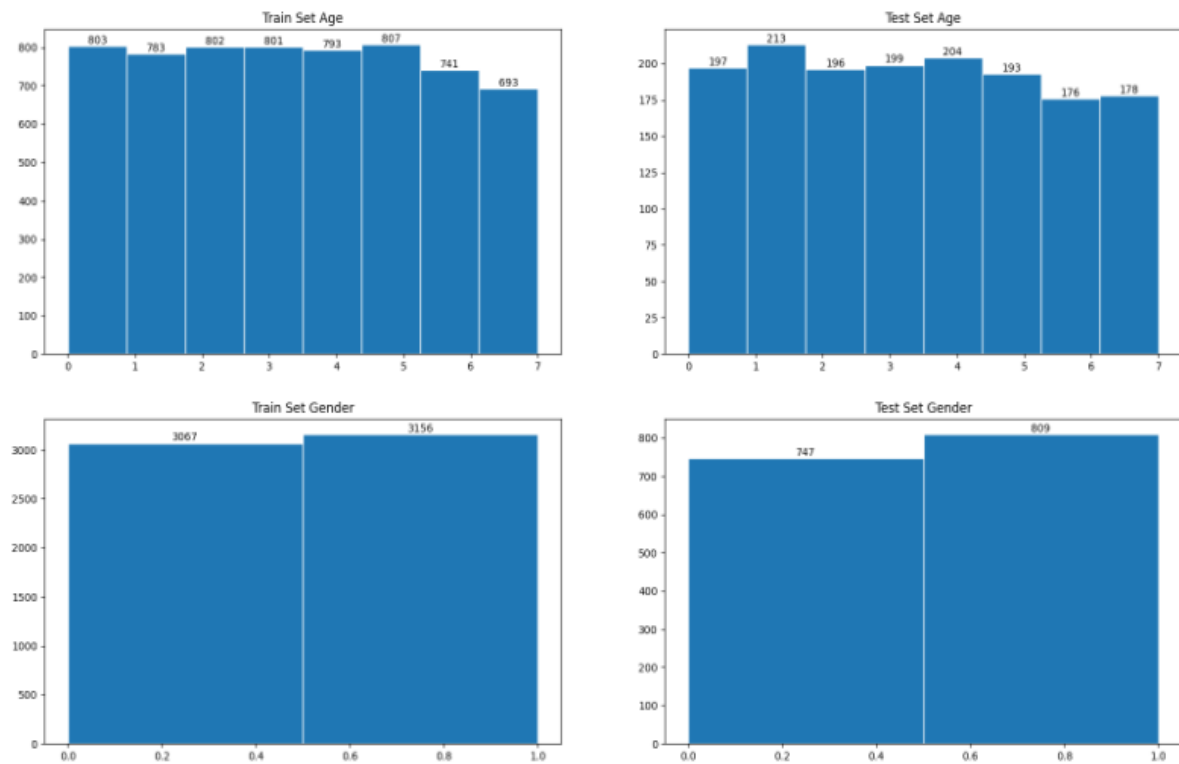


Figure 12: Train & Test split

- We have created two models, first is using SVM algorithm and second is by using CNN model.
- The features we have used for detecting Age and gender of human from images & live video feed are :
  - **Gender classification :**
    - GEOMETRIC BASED FEATURE EXTRACTION :
      - describe shape and location of facial components.
      - Find the coordinates of these facial points.
    - APPEARANCE BASED FEATURE EXTRACTION.

- **Age classification :**

- **WRINKLE FEATURES :**

- estimate of the F5 characteristics can be carried out.
- as age increases , wrinkles on face turn out to be clearer.

- **GEOMETRIC FEATURES :**

- It is based on calculating ratios between different measurements of facial features( e.g., Eyes, nose, mouth, chin, etc.)
- distance values ratios .

- a) **SVM Model :**

- Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.
- The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.
- SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

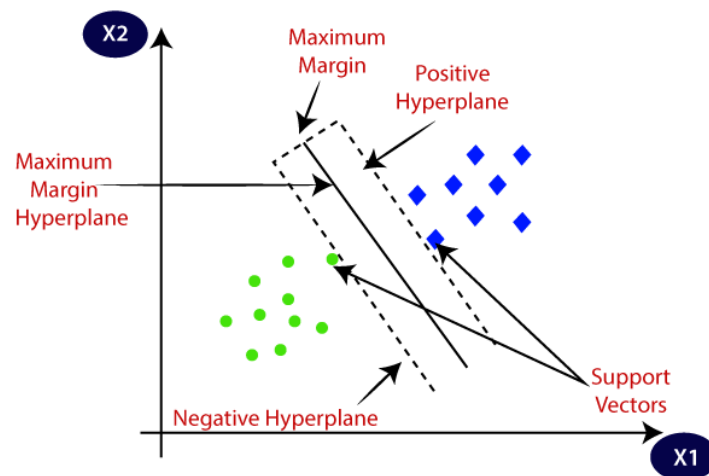


Figure 13: SVM



### Steps for SVM age and gender model

- As the dataset has been prepared, now it's the time for training different classifiers and then choose the best one. We will be training classifiers with hyperparameter tuning. The performance of a model significantly depends on the value of hyperparameters.
- GridSearchCV is the name of the package provided by Scikit-learn which is then used for hyperparameter tuning.
- GridSearchCV is the process of performing hyperparameter tuning in order to determine the optimal values for a given model. Doing this manually could take a considerable amount of time and resources and thus we use GridSearchCV to automate the tuning of hyperparameters.
- In our project we have created two different models, one for age and other for gender. Because SVM doesn't support multiclass classification natively. It supports binary classification.
- At last Model is saved and then the RAM is cleared. Now, the performance on the test dataset takes place.

### **b) CNN :**

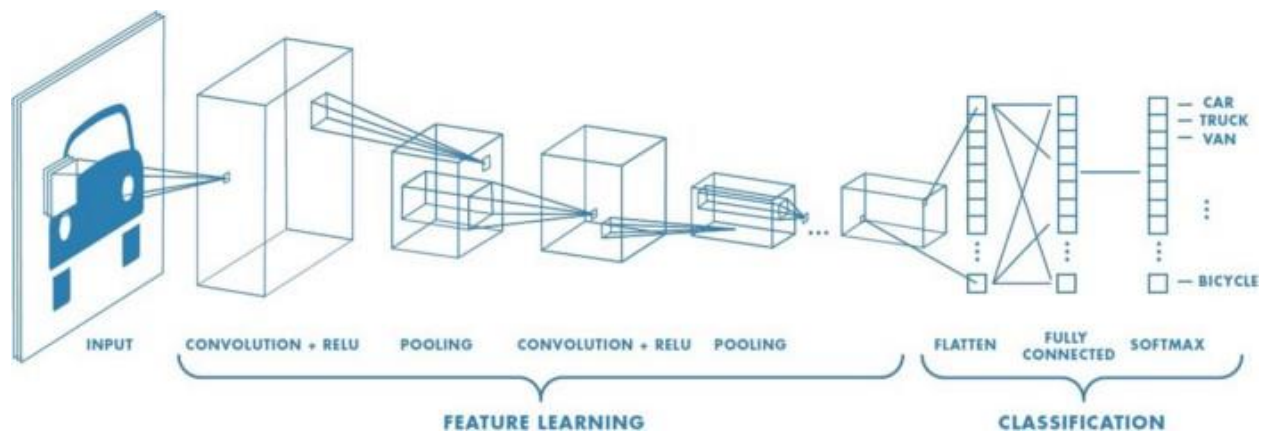


Figure 14: CNN Architecture 1

- A **Convolutional Neural Network (ConvNet/CNN)** is a Deep Learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms, because while in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.
- The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlaps to cover the entire visual area.
- An image is nothing but a matrix of pixel values, right? So why not just flatten the image (e.g. 3x3 image matrix into a 9x1 vector) and feed it to a Multi-Level Perceptron for classification purposes? In cases of extremely basic binary images, the method might show an average precision score while performing prediction of classes but would have little to no accuracy when it comes to complex images having pixel dependencies throughout.

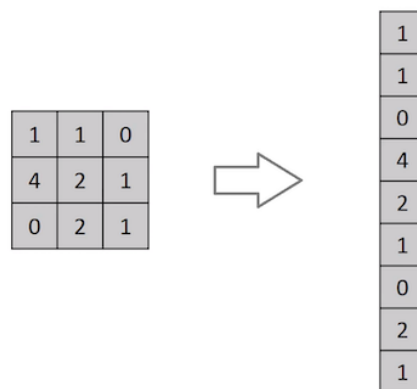


Figure 15: Flattening of a 3x3 image matrix into a 9x1 vector

- A ConvNet is able to **successfully capture the Spatial and Temporal dependencies** in an image through the application of relevant filters. The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and the reusability of weights. In other words, the network can be trained to understand the sophistication of the image better.

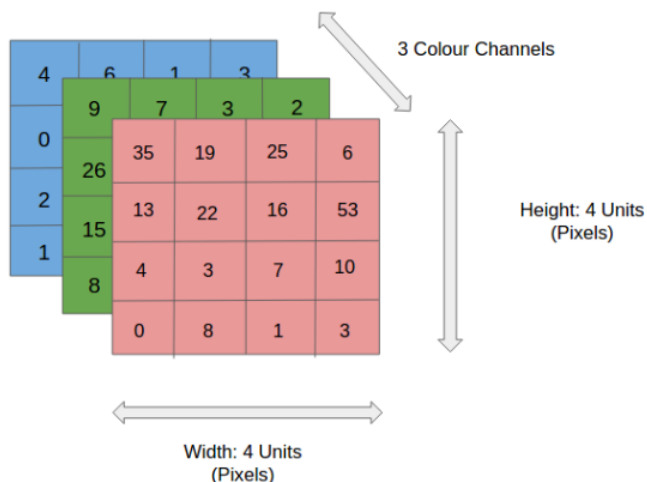


Figure 16: 4x4x3 RGB Image

- In this figure, we have an RGB image that has been separated by its three-color planes — Red, Green, and Blue. There are a number of such color spaces in which images exist — Grayscale, RGB, HSV, CMYK, etc.
- Thus it becomes computationally intensive things would get once the images reach dimensions, say 8K (7680×4320). The role of ConvNet is to reduce the images into a form that is easier to process, without losing features that are critical for getting a good prediction. This is important when we are to design an architecture that is not only good at learning features but also scalable to massive datasets.

- CNN is specifically designed to process input images. Their architecture is then more specific: it is composed of two main blocks.

**The first block** makes the particularity of this type of neural network since it functions as a feature extractor. To do this, it performs template matching by applying convolution filtering operations. The first layer filters the image with several convolution kernels and returns “**feature maps**”, which are then normalized (with an activation function) and/or resized.

This process can be repeated several times: we filter the features maps obtained with new kernels, which gives us new features maps to normalize and resize, and we can filter again, and so on. Finally, the values of the last feature maps are concatenated into a vector. This vector defines the output of the first block and the input of the second.

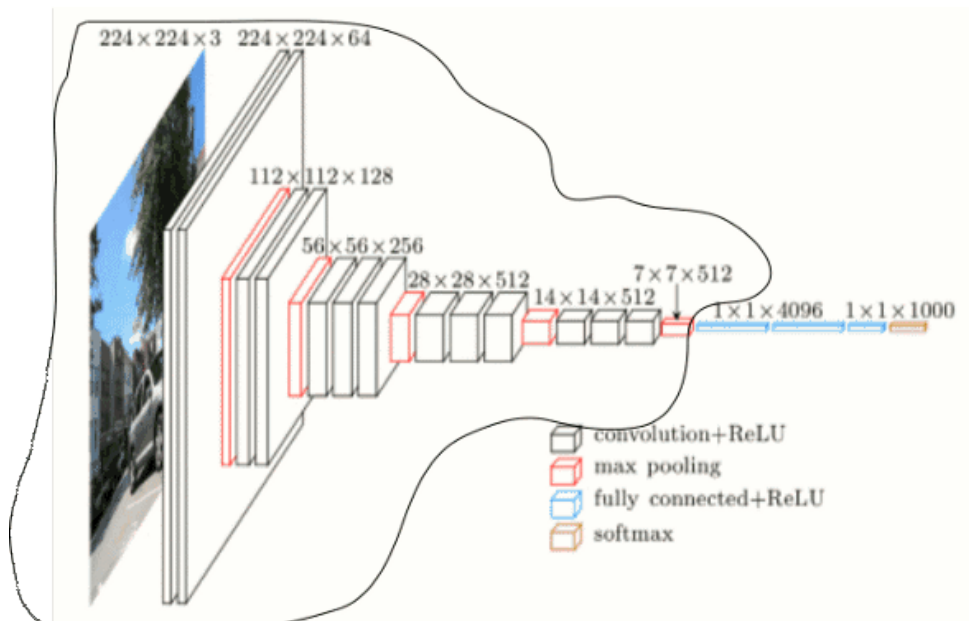


Figure 17: The first block is encircled in black

**The second block** is not characteristic of a CNN: it is in fact at the end of all the neural networks used for classification. The input vector values are transformed (with several linear combinations and activation functions) to return a new vector to the output. This last vector contains as many elements as there are classes: element  $i$  represents the probability that the image belongs to class  $i$ . Each element is therefore between 0 and 1, and the sum of all is worth 1. These probabilities are calculated by the last layer of this block (and therefore of the network), which uses a **logistic function** (binary classification) or a **softmax function** (multi-class classification) as an activation function.

As with ordinary neural networks, the parameters of the layers are determined by gradient backpropagation: the **cross-entropy** is minimized during the training phase. But in the case of CNN, these parameters refer in particular to the image features.

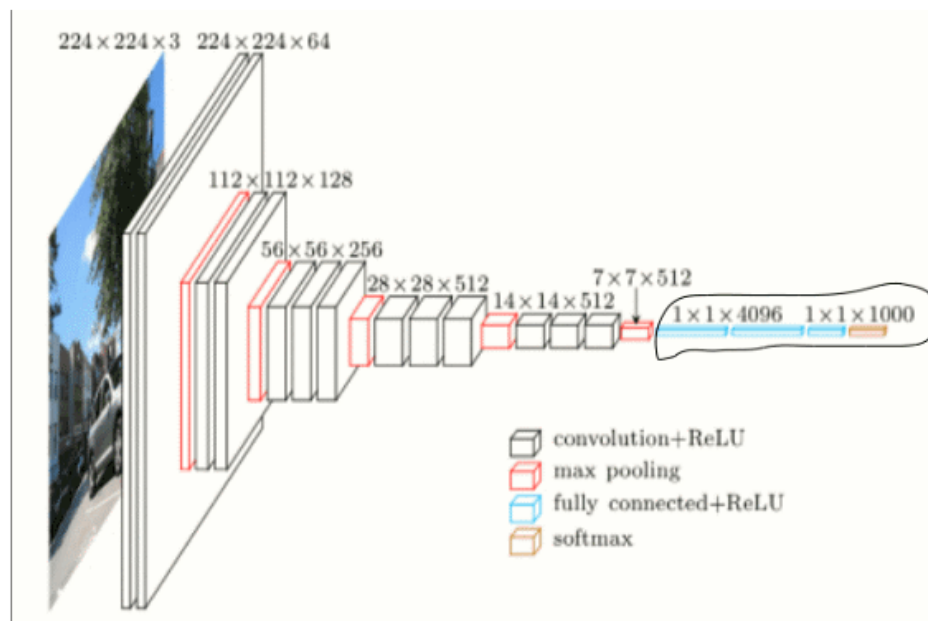


Figure 18: The second block is encircled in black

## CNN Layers:

There are three types of layers that make up the CNN which are the convolutional layers, pooling layers, and fully-connected (FC) layers. When these layers are stacked, a CNN architecture will be formed. In addition to these three layers, there are two more important parameters which are the dropout layer and the activation function which are defined below.

### 1. Convolutional Layer

- This layer is the first layer that is used to extract the various features from the input images. In this layer, the mathematical operation of convolution is performed between the input image and a filter of a particular size  $M \times M$ .
- By sliding the filter over the input image, the dot product is taken between the filter and the parts of the input image with respect to the size of the filter ( $M \times M$ ).
- The output is termed as the Feature map which gives us information about the image such as the corners and edges. Later, this feature map is fed to other layers to learn several other features of the input image.
- The convolution layer in CNN passes the result to the next layer once applying the convolution operation in the input. Convolutional layers in CNN benefit a lot as they ensure the spatial relationship between the pixels is intact.

### 2. Pooling Layer

- In most cases, a Convolutional Layer is followed by a Pooling Layer.
- The primary aim of this layer is to decrease the size of the convolved feature map to reduce the computational costs.
- This is performed by decreasing the connections between layers and independently operates on each feature map.

- Depending upon method used, there are several types of Pooling operations. It basically summarises the features generated by a convolution layer.
  - In Max Pooling, the largest element is taken from feature map.
  - Average Pooling calculates the average of the elements in a predefined sized Image section. The total sum of the elements in the predefined section is computed in Sum Pooling.
- The Pooling Layer usually serves as a bridge between the Convolutional Layer and the FC Layer.
- This CNN model generalises the features extracted by the convolution layer, and helps the networks to recognise the features independently. With the help of this, the computations are also reduced in a network.

### 3. Fully Connected Layer

- The Fully Connected (FC) layer consists of the weights and biases along with the neurons and is used to connect the neurons between two different layers.
- These layers are usually placed before the output layer and form the last few layers of a CNN Architecture.
- In this, the input image from the previous layers is flattened and fed to the FC layer.
- The flattened vector then undergoes few more FC layers where the mathematical functions operations usually take place. In this stage, the classification process begins to take place. The reason two layers are connected is that two fully connected layers will perform better than a single connected layer. These layers in CNN reduce the human supervision.

#### 4. Dropout

- Usually, when all the features are connected to the FC layer, it can cause overfitting in the training dataset.
- Overfitting occurs when a particular model works so well on the training data causing a negative impact in the model's performance when used on a new data.
- To overcome this problem, a dropout layer is utilised wherein a few neurons are dropped from the neural network during training process resulting in reduced size of the model. On passing a dropout of 0.3, 30% of the nodes are dropped out randomly from the neural network.
- Dropout results in improving the performance of a machine learning model as it prevents overfitting by making the network simpler. It drops neurons from the neural networks during training.

#### 5. Activation Functions

- Finally, one of the most important parameters of the CNN model is the activation function. They are used to learn and approximate any kind of continuous and complex relationship between variables of the network.
- In simple words, it decides which information of the model should fire in the forward direction and which ones should not at the end of the network.
- It adds non-linearity to the network.
- There are several commonly used activation functions such as the ReLU, Softmax, tanH and the Sigmoid functions.
- Each of these functions have a specific usage. For a binary classification CNN model, sigmoid and softmax functions are preferred and for a multi-class classification, generally softmax is used. In simple terms, activation functions in a CNN model determine whether a neuron should be activated or not. It decides whether the input to the work is important or not to predict using mathematical operations.



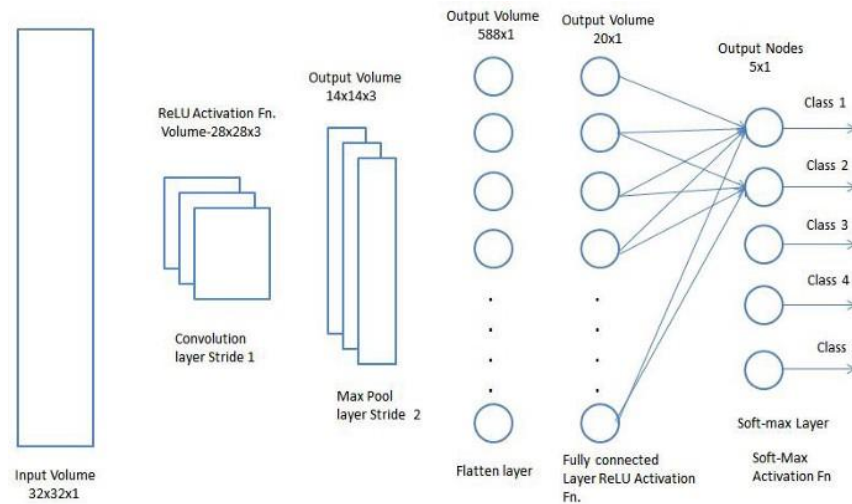


Figure 19: CNN architecture 2

- We have made CNN model for age and gender i.e.; it is a multi-output model.
- The CNN architecture contains two branches, one for age, other for gender. Each branch contains a sequence of Convolutional Layers.
- The structure of our network :
  - The input shape of our network is :  $\text{INPUT\_SHAPE} = (196, 196, 3)$  First, it contains a default set of hidden layers, which is called by “backbone function”.
  - The structure used in this network is defined as:

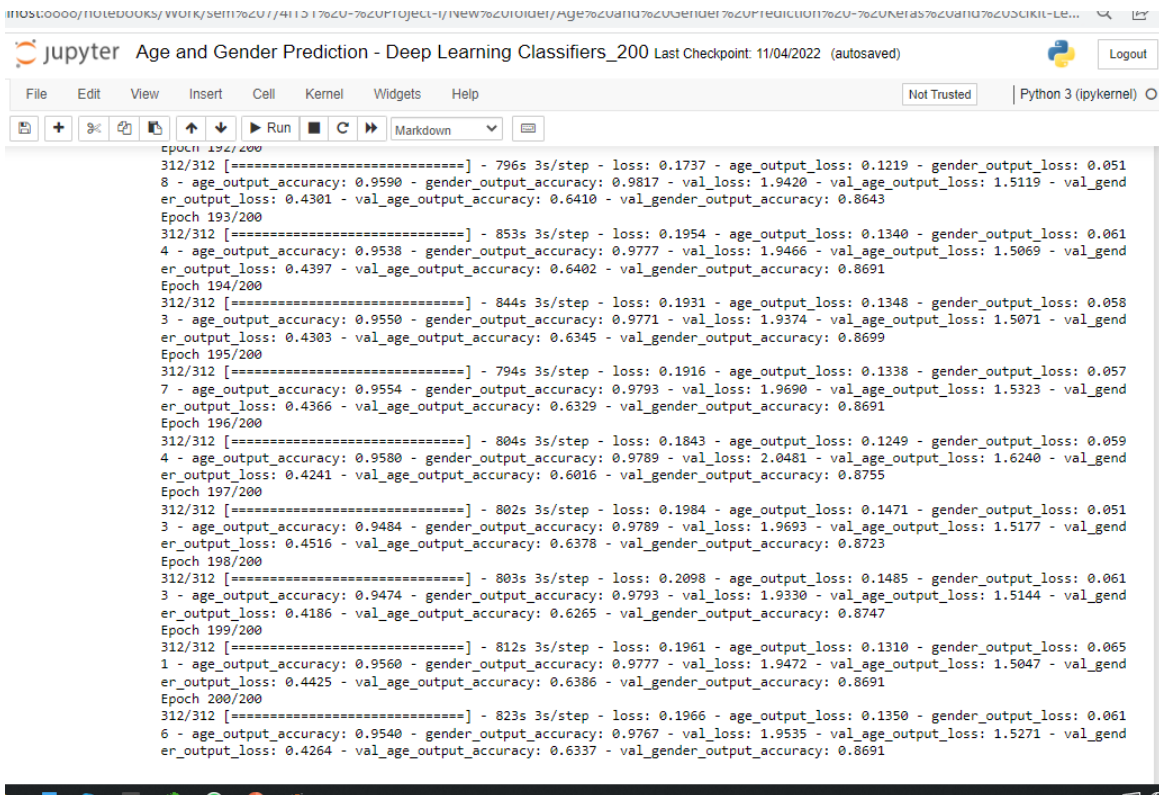
Conv2D -> Batch Normalization -> Pooling -> Dropout

- Now, this default set output is connected to age and gender network.
- The age and gender network uses activation “sigmoid”. At last, “build function” is used to assemble our multi-output model CNN.

### 3.3.5 Model Training

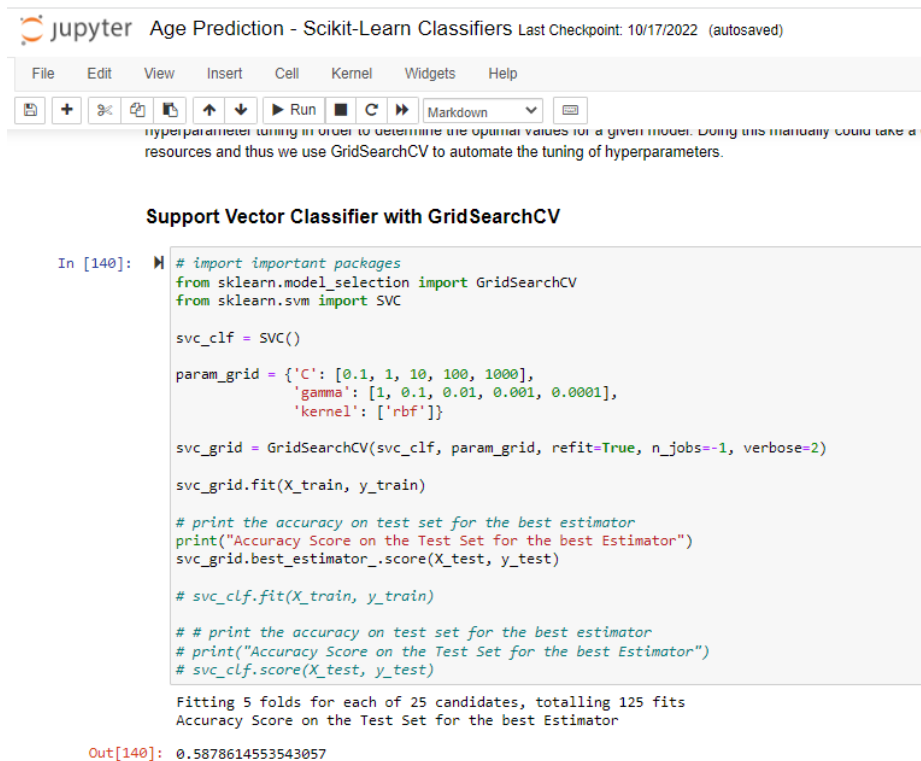
- Once the model is built, needs to be trained. Some parameters like optimizer, compile the model, setting the learning rate were defined and also data augmentation was performed. Then, epochs and batch size were defined that helped in training the model.

#### 1. Age & Gender training for CNN model :



```
Epoch 192/200
312/312 [=====] - 796s 3s/step - loss: 0.1737 - age_output_loss: 0.1219 - gender_output_loss: 0.0518 - age_output_accuracy: 0.9590 - gender_output_accuracy: 0.9817 - val_loss: 1.9420 - val_age_output_loss: 1.5119 - val_gender_output_loss: 0.4301 - val_age_output_accuracy: 0.6410 - val_gender_output_accuracy: 0.8643
Epoch 193/200
312/312 [=====] - 853s 3s/step - loss: 0.1954 - age_output_loss: 0.1340 - gender_output_loss: 0.0614 - age_output_accuracy: 0.9538 - gender_output_accuracy: 0.9777 - val_loss: 1.9466 - val_age_output_loss: 1.5069 - val_gender_output_loss: 0.4397 - val_age_output_accuracy: 0.6402 - val_gender_output_accuracy: 0.8691
Epoch 194/200
312/312 [=====] - 844s 3s/step - loss: 0.1931 - age_output_loss: 0.1348 - gender_output_loss: 0.0583 - age_output_accuracy: 0.9550 - gender_output_accuracy: 0.9771 - val_loss: 1.9374 - val_age_output_loss: 1.5071 - val_gender_output_loss: 0.4303 - val_age_output_accuracy: 0.6345 - val_gender_output_accuracy: 0.8699
Epoch 195/200
312/312 [=====] - 794s 3s/step - loss: 0.1916 - age_output_loss: 0.1338 - gender_output_loss: 0.0577 - age_output_accuracy: 0.9554 - gender_output_accuracy: 0.9793 - val_loss: 1.9690 - val_age_output_loss: 1.5323 - val_gender_output_loss: 0.4366 - val_age_output_accuracy: 0.6329 - val_gender_output_accuracy: 0.8691
Epoch 196/200
312/312 [=====] - 804s 3s/step - loss: 0.1843 - age_output_loss: 0.1249 - gender_output_loss: 0.0594 - age_output_accuracy: 0.9580 - gender_output_accuracy: 0.9789 - val_loss: 2.0481 - val_age_output_loss: 1.6240 - val_gender_output_loss: 0.4241 - val_age_output_accuracy: 0.6016 - val_gender_output_accuracy: 0.8755
Epoch 197/200
312/312 [=====] - 802s 3s/step - loss: 0.1984 - age_output_loss: 0.1471 - gender_output_loss: 0.0513 - age_output_accuracy: 0.9484 - gender_output_accuracy: 0.9789 - val_loss: 1.9693 - val_age_output_loss: 1.5177 - val_gender_output_loss: 0.4516 - val_age_output_accuracy: 0.6378 - val_gender_output_accuracy: 0.8723
Epoch 198/200
312/312 [=====] - 803s 3s/step - loss: 0.2098 - age_output_loss: 0.1485 - gender_output_loss: 0.0613 - age_output_accuracy: 0.9474 - gender_output_accuracy: 0.9793 - val_loss: 1.9330 - val_age_output_loss: 1.5144 - val_gender_output_loss: 0.4186 - val_age_output_accuracy: 0.6265 - val_gender_output_accuracy: 0.8747
Epoch 199/200
312/312 [=====] - 812s 3s/step - loss: 0.1961 - age_output_loss: 0.1310 - gender_output_loss: 0.0651 - age_output_accuracy: 0.9560 - gender_output_accuracy: 0.9777 - val_loss: 1.9472 - val_age_output_loss: 1.5047 - val_gender_output_loss: 0.4425 - val_age_output_accuracy: 0.6386 - val_gender_output_accuracy: 0.8691
Epoch 200/200
312/312 [=====] - 823s 3s/step - loss: 0.1966 - age_output_loss: 0.1350 - gender_output_loss: 0.0616 - age_output_accuracy: 0.9540 - gender_output_accuracy: 0.9767 - val_loss: 1.9535 - val_age_output_loss: 1.5271 - val_gender_output_loss: 0.4264 - val_age_output_accuracy: 0.6337 - val_gender_output_accuracy: 0.8691
```

## 2. Age training for SVM model :



The image shows a Jupyter Notebook titled "Age Prediction - Scikit-Learn Classifiers". The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running cells, and markdown editing. The notebook content consists of a code cell with the following Python code:

```
In [140]: # import important packages
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVC

svc_clf = SVC()

param_grid = {'C': [0.1, 1, 10, 100, 1000],
              'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
              'kernel': ['rbf']}

svc_grid = GridSearchCV(svc_clf, param_grid, refit=True, n_jobs=-1, verbose=2)

svc_grid.fit(X_train, y_train)

# print the accuracy on test set for the best estimator
print("Accuracy Score on the Test Set for the best Estimator")
svc_grid.best_estimator_.score(X_test, y_test)

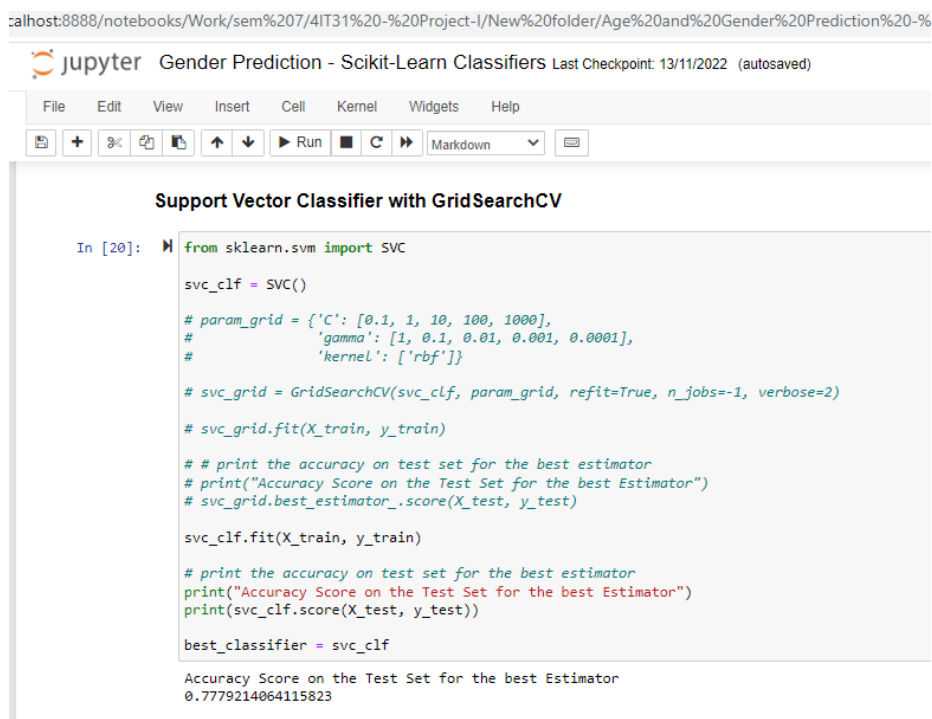
# svc_clf.fit(X_train, y_train)

# # print the accuracy on test set for the best estimator
# print("Accuracy Score on the Test Set for the best Estimator")
# svc_clf.score(X_test, y_test)

Fitting 5 folds for each of 25 candidates, totalling 125 fits
Accuracy Score on the Test Set for the best Estimator

Out[140]: 0.5878614553543057
```

## 3. Gender training for SVM model :



The image shows a Jupyter Notebook titled "Gender Prediction - Scikit-Learn Classifiers". The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running cells, and markdown editing. The notebook content consists of a code cell with the following Python code:

```
In [20]: from sklearn.svm import SVC

svc_clf = SVC()

# param_grid = {'C': [0.1, 1, 10, 100, 1000],
#               'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
#               'kernel': ['rbf']}

# svc_grid = GridSearchCV(svc_clf, param_grid, refit=True, n_jobs=-1, verbose=2)

# svc_grid.fit(X_train, y_train)

# # print the accuracy on test set for the best estimator
# print("Accuracy Score on the Test Set for the best Estimator")
# svc_grid.best_estimator_.score(X_test, y_test)

svc_clf.fit(X_train, y_train)

# print the accuracy on test set for the best estimator
print("Accuracy Score on the Test Set for the best Estimator")
print(svc_clf.score(X_test, y_test))

best_classifier = svc_clf

Accuracy Score on the Test Set for the best Estimator
0.7779214064115823
```

### 3.3.6 Model Evaluation

- The testing accuracy and validation accuracy of our model are checked, model loss is noted, and the confusion matrix is plotted. The misclassified images count of each type is also determined.

- **Training Loss vs Validation Loss Plot**

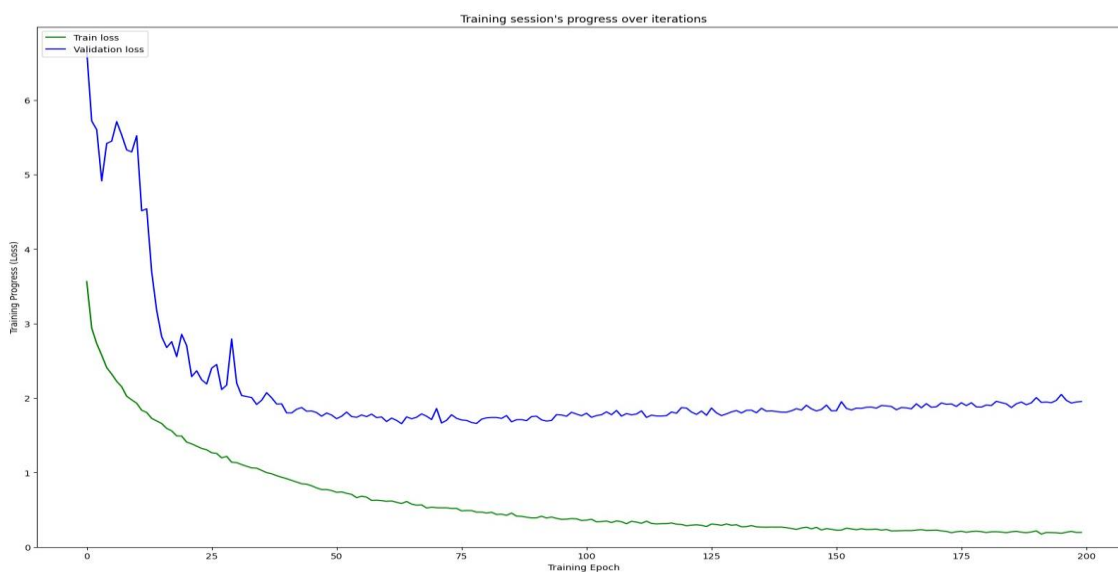


Figure 20: Training Loss vs Validation Loss Plot

- **Training Accuracy Vs Validation Accuracy for Age**

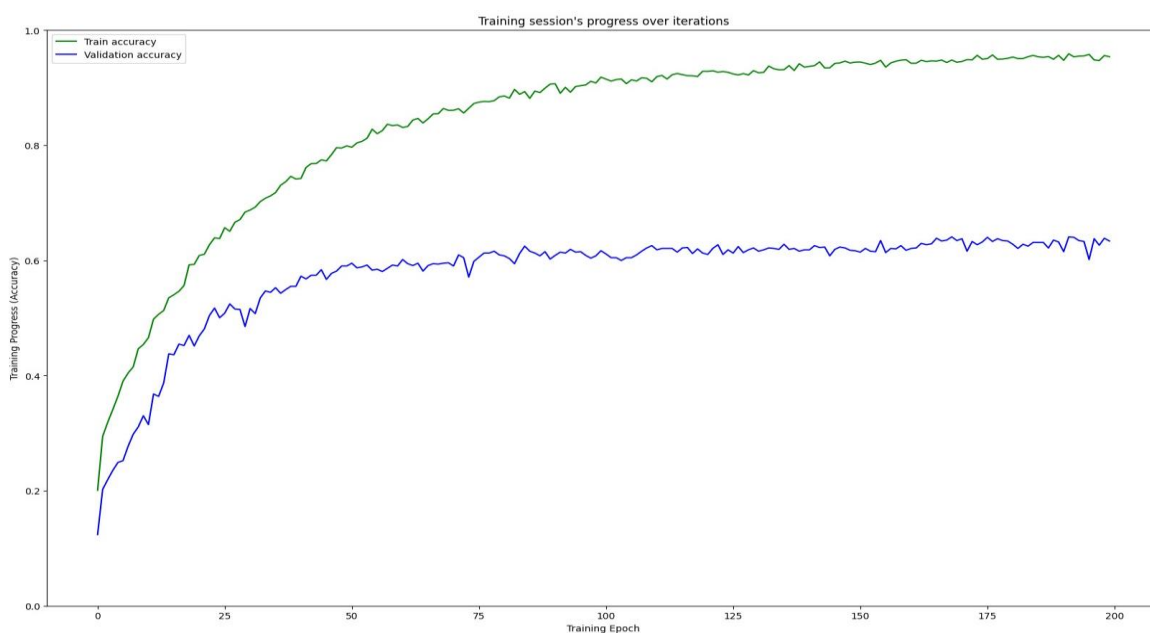


Figure 21: Training Accuracy vs Validation Accuracy for Age

- **Training Accuracy Vs Validation Accuracy for Gender**

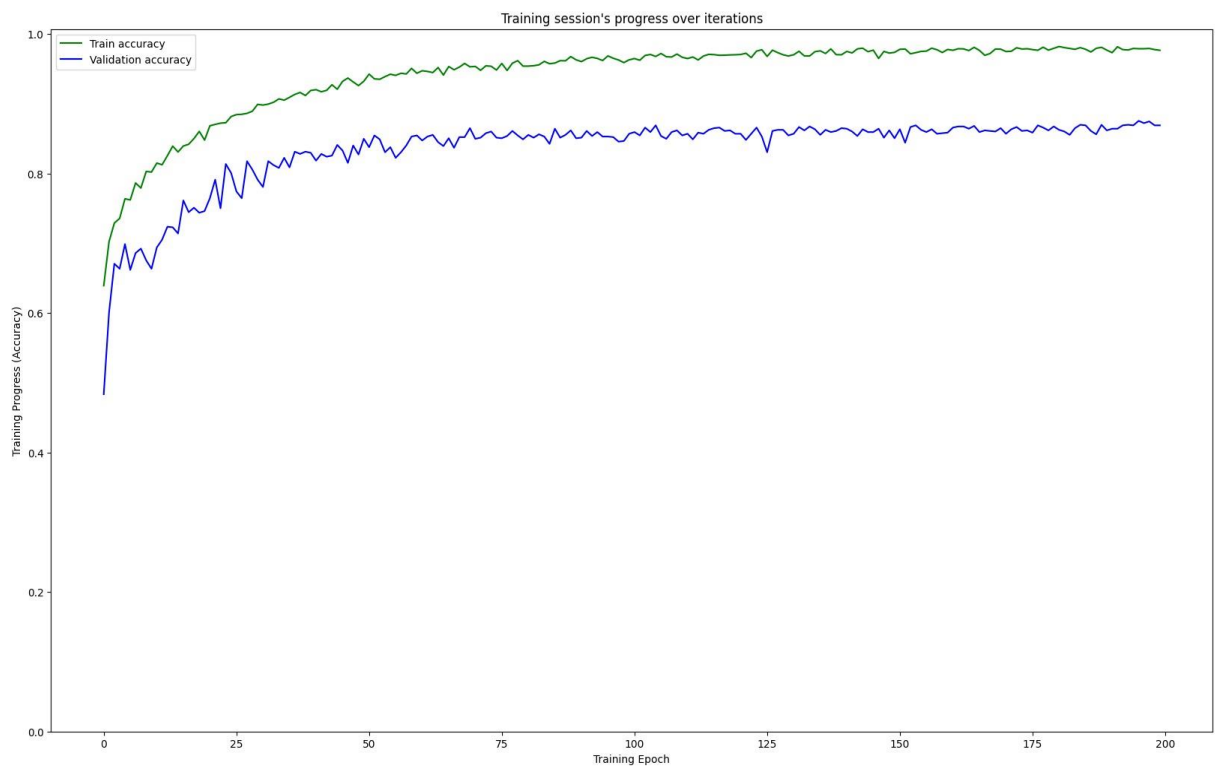


Figure 22: Training Accuracy vs Validation Accuracy for Gender

**CONFUSION MATRIX :**



Figure 23: Confusion Matrix of SVM age



Figure 24: Confusion Matrix of SVM gender

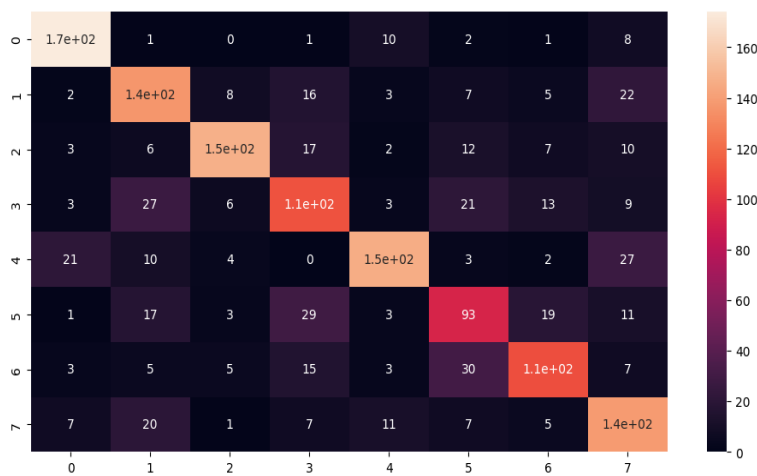


Figure 25: Confusion Matrix of CNN age

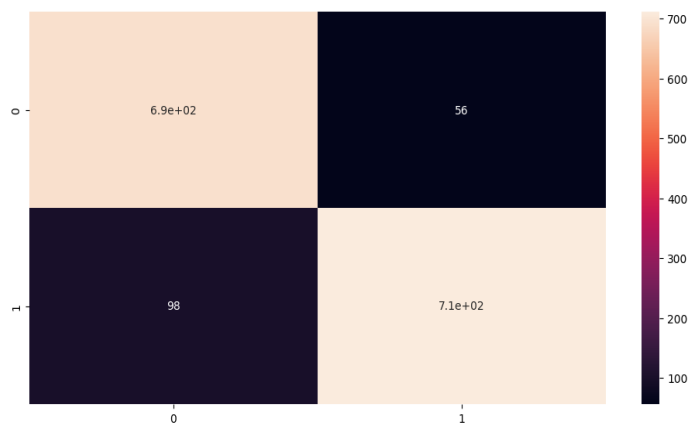


Figure 26: Confusion Matrix of CNN gender

### 3.4 Project SRS

#### 3.4.1 USE CASE DIAGRAM

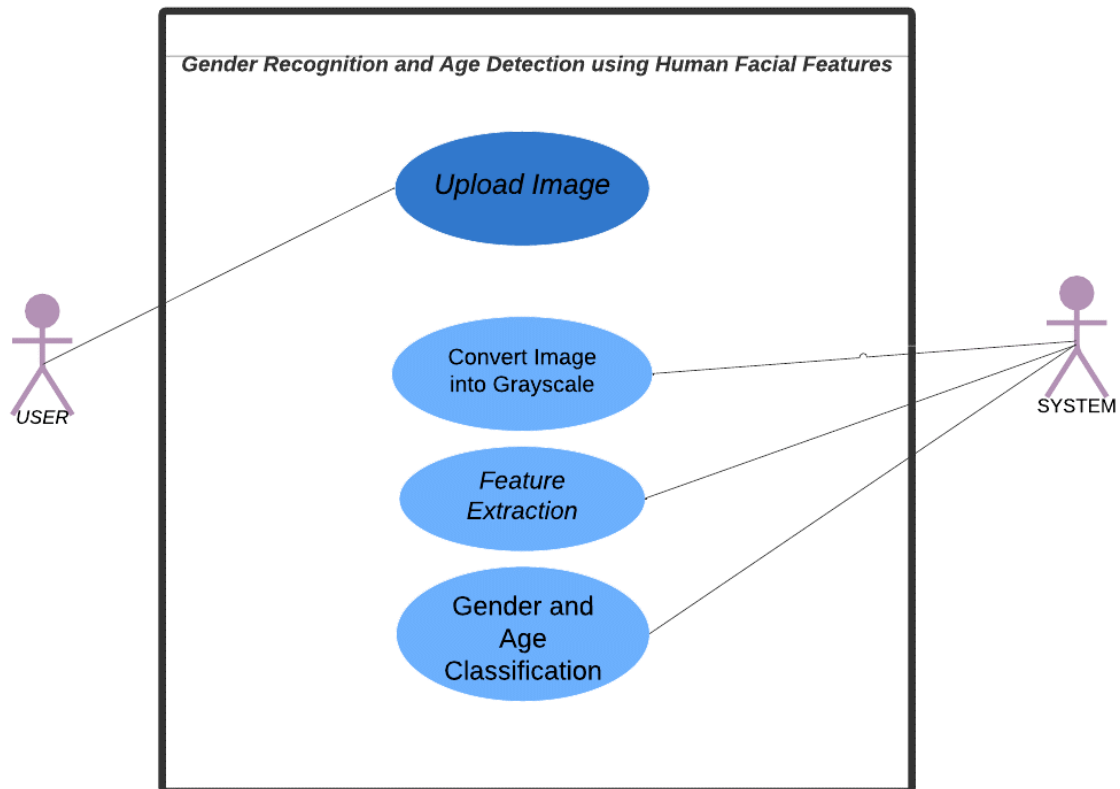


Figure 27: Use Case Diagram

### 3.4.2 ER DIAGRAM

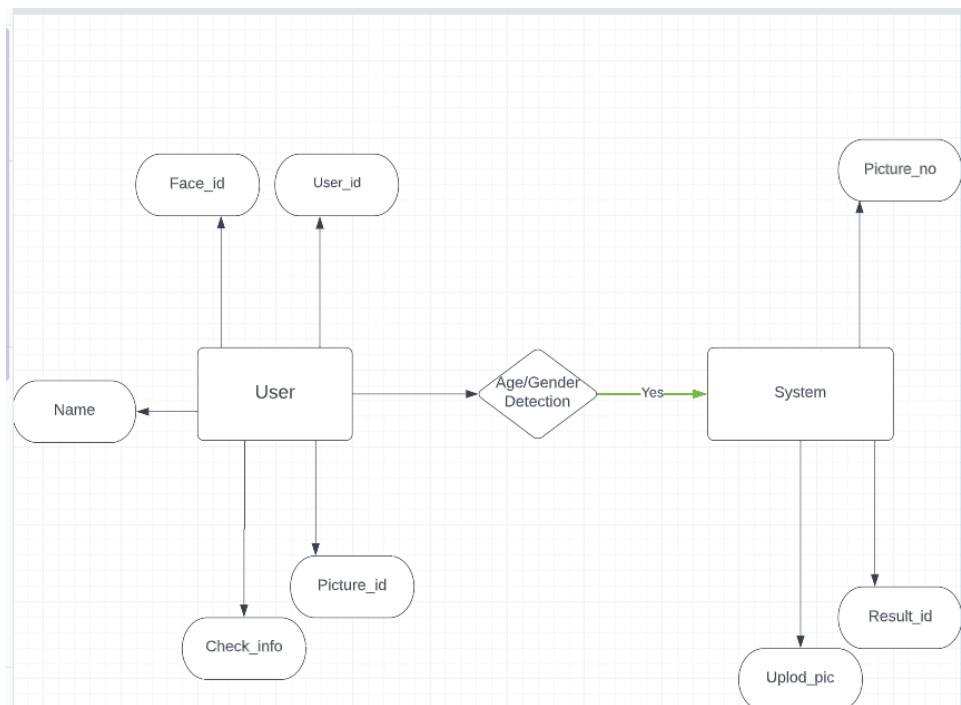


Figure 28: ER Diagram

### 3.4.3 SEQUENCE DIAGRAM

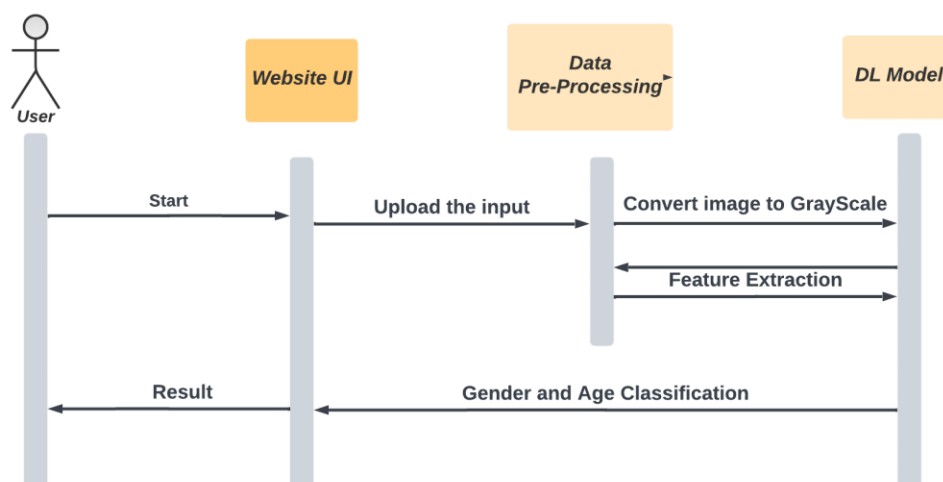


Figure 29: Sequence Diagram



### 3.4.4 DATA FLOW DIAGRAM

#### Level 0:

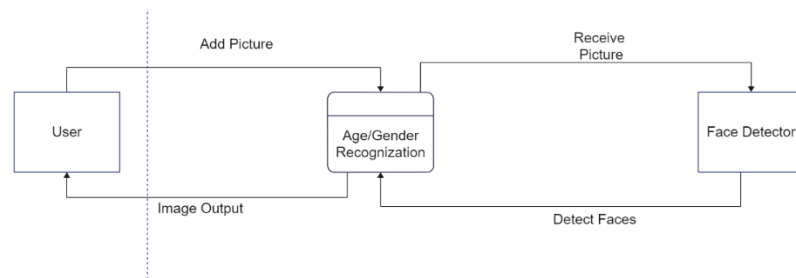


Figure 30: Data Flow Diagram(Level 0)

#### Level 1:

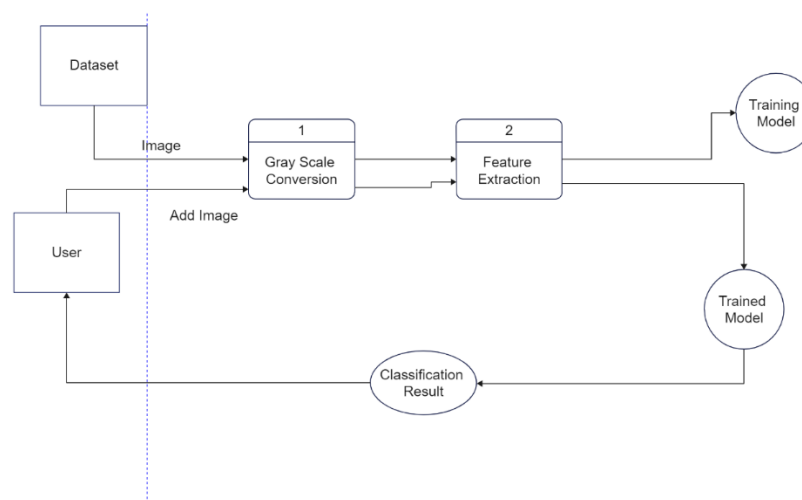


Figure 31: Data Flow Diagram(Level 1)

## Chapter 4 : Implementation & Testing

### 4.1 User Interface and Snapshot

- User Interface is a point of human computer interaction and communication in a device. UI should be simple and user friendly. User should not face any problem using UI.
  - We have made simple UI using Flask framework in python
  - User can select SVM or CNN model that we have trained for detecting Age and Gender in live video feed.
  - Flask is a micro web framework, used for developing web applications.
  - It serves as a foundation for software developers, allowing them to create a variety of applications for certain platforms. It is a set of functions and predefined classes used to connect with the system software and handle inputs and outputs.
  - To Run our project :
1. Open cmd prompt, and see if you have conda installed.
    - conda is a tool for managing and deploying applications, environments and packages.
  2. If you have conda preinstalled than run this code for creating our virtual environment named “TF”, with python version 3.7.

**conda create -n TF python==3.7**

3. After creating our environment, we activate it, and install all required packages in it. To activate our virtual environment, the code is :

**conda activate TF**

4. To run our flask web app, select the folder where our flask code as well as virtual environment is there, and write this code:

### conda activate TF

```

C:\Windows\System32\cmd.exe - python run.py
Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP\Work\sem 7\4IT31 - Project-I\New folder\Age and Gender Prediction - Keras and Scikit-Learn\Age and Gender Prediction - Keras and Scikit-Learn\Flask>conda activate TF

(TF) C:\Users\HP\Work\sem 7\4IT31 - Project-I\New folder\Age and Gender Prediction - Keras and Scikit-Learn\Age and Gender Prediction - Keras and Scikit-Learn\Flask>python run.py
2022-12-16 03:36:03.181769: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cudart64_110.dll'; dlderror: cudart64_110.dll not found
2022-12-16 03:36:03.184953: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
2022-12-16 03:36:35.891112: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'nvcuda.dll'; dlerror: nvcuda.dll not found
2022-12-16 03:36:35.892719: W tensorflow/stream_executor/cuda/cuda_driver.cc:263] failed call to cuInit: UNKNOWN ERROR (303)
2022-12-16 03:36:35.911009: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-SBQULBV
2022-12-16 03:36:35.911558: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-SBQULBV
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://192.168.27.222:5000
Press CTRL+C to quit
* Restarting with stat
2022-12-16 03:36:46.533602: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cudart64_110.dll'; dlerror: cudart64_110.dll not found
2022-12-16 03:36:46.533973: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not

```

### python run.py

```

C:\Windows\System32\cmd.exe - python run.py
Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP\Work\sem 7\4IT31 - Project-I\New folder\Age and Gender Prediction - Keras and Scikit-Learn\Age and Gender Prediction - Keras and Scikit-Learn\Flask>conda activate TF

(TF) C:\Users\HP\Work\sem 7\4IT31 - Project-I\New folder\Age and Gender Prediction - Keras and Scikit-Learn\Age and Gender Prediction - Keras and Scikit-Learn\Flask>python run.py
2022-12-16 03:36:03.181769: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cudart64_110.dll'; dlerror: cudart64_110.dll not found
2022-12-16 03:36:03.184953: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
2022-12-16 03:36:35.891112: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'nvcuda.dll'; dlerror: nvcuda.dll not found
2022-12-16 03:36:35.892719: W tensorflow/stream_executor/cuda/cuda_driver.cc:263] failed call to cuInit: UNKNOWN ERROR (303)
2022-12-16 03:36:35.911009: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-SBQULBV
2022-12-16 03:36:35.911558: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-SBQULBV
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://192.168.27.222:5000
Press CTRL+C to quit
* Restarting with stat
2022-12-16 03:36:46.533602: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cudart64_110.dll'; dlerror: cudart64_110.dll not found
2022-12-16 03:36:46.533973: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not

```

5. Our Flask web is now running on our localhost whose URL is given as:

**http://127.0.0.1:5000**

```

C:\Users\HP\Work\sem 7\4IT31 - Project-I\New folder\Age and Gender Prediction - Keras and Scikit-Learn\Flask>conda activate TF
(TF) C:\Users\HP\Work\sem 7\4IT31 - Project-I\New folder\Age and Gender Prediction - Keras and Scikit-Learn\Flask>python run.py
2022-12-16 03:36:03.181769: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cudart64_110.dll'; dlderror: cudart64_110.dll not found
2022-12-16 03:36:03.184953: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
2022-12-16 03:36:35.891112: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'nvcuda.dll'; dlderror: nvcuda.dll not found
2022-12-16 03:36:35.892719: W tensorflow/stream_executor/cuda/cuda_driver.cc:263] failed call to cuInit: UNKNOWN ERROR (303)
2022-12-16 03:36:35.911009: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-SBQULBV
2022-12-16 03:36:35.911558: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-SBQULBV
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://192.168.27.222:5000
Press CTRL+C to quit
* Restarting with stat
2022-12-16 03:36:46.533602: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cudart64_110.dll'; dlderror: cudart64_110.dll not found
2022-12-16 03:36:46.533973: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
```

6. Home Page of our Flask web app:

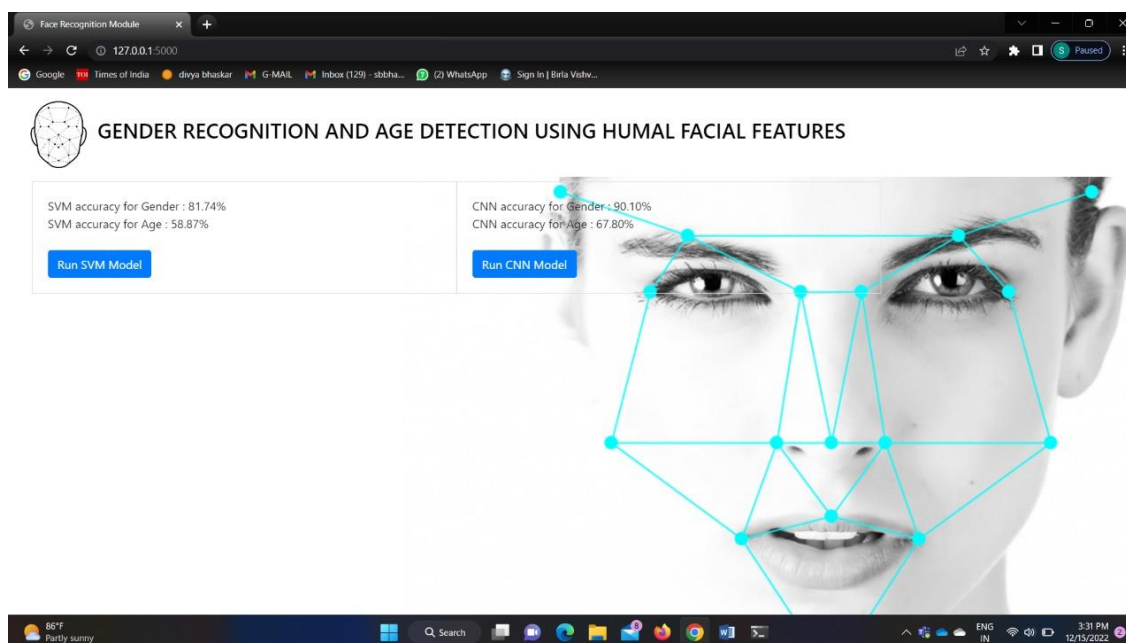


Figure 32: User Interface of Home Page

## 4.2 Testing using Use Case

### 1. CNN model output :



Figure 33: Output of CNN Model

### 2. SVM model output :

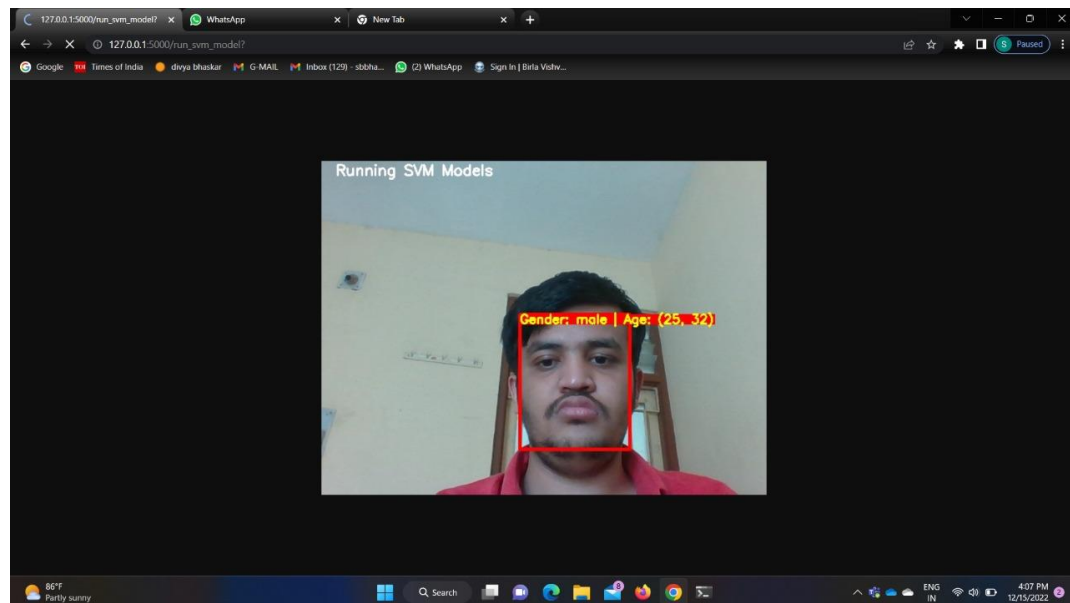


Figure 34: Output of SVM Model

## Chapter 5 : Conclusion & Future Work

- The Result obtained in our project is :
- Accuracy for CNN model for Gender is 90.10% and for Age it is 67.80%. whereas,
- Accuracy for SVM model for Gender is 81.74% and for Age it is 60%.
- Thus, two important conclusions can be made from our results:
  - First, CNN can be used to provide improved age and gender classification results, even considering the much smaller size of contemporary unconstrained image sets labeled for age and gender.
  - Second, the simplicity of our model implies that more elaborate systems using more training data may well be capable of substantially improving results beyond those reported here.
- The future work that we can do for our project to improve it are :
  - We can add more images in our dataset to improve the accuracy.
  - We can incorporate this system which helps in finding perfect ads on social media applications based on human faces.

## Chapter 6 : References

- [1] Dr.c.k.gomathy, a.lokesh, ch.harshavardhanreddy and a.sai kiran, “Age And Gender Detection”, 2021, International Journal of Scientific Research in Engineering and Management(IJSREM).
- [2] Miss Vaishnavi mali and Dr.Babasaheb G. Patil, “Human gender, classification using machine learning”,2020, International Journal of Engineering Research and Technology (IJERT).
- [3] Syed Taskeen Rahman, Asiful Arefeen, Shashoto Sharif Mridul, Asir Intisar Khan and Samia Subrina, “Human Age and Gender Estimation using Facial Image Processing”, 2020, IEEE.
- [4] Ishita Verma,Urvi Marhatta, Sachin Sharma and Vijay Kumar, “Age Prediction using Image Dataset using Machine Learning”, 2019, International Journal of Innovative Technology and Exploring Engineering (IJITEE).
- [5] Rahul Kumar, Akhilesh Kumar Srivastava and Deepak Kumar Agarwal, “Estimation Of Age Group based on Facial Features”, 2018, International Journal of Engineering Research & Technology (IJERT).
- [6] Sonu Dhall and Poonam Sethi, “Geometric and appearance feature analysis for facial expression recognition”,2014, International Journal of Advanced Engineering Technology.
- [7] Gil Levi and Tal Hassner, “Age and Gender Classification using Convolutional Neural Networks”, 2015, The Open University of Israel.
- [8] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Senior Member, IEEE, and Yu Qiao, Senior Member, IEEE, “Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks”, 2016, IEEE Signal Processing Letters.

[9] A comprehensive guide to convolutional neural networks:

<https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>

[10] Understand the architecture of CNN:

<https://towardsdatascience.com/understand-the-architecture-of-cnn-90a25e244c7>

[11] Basic CNN architecture:

<https://www.upgrad.com/blog/basic-cnn-architecture/>