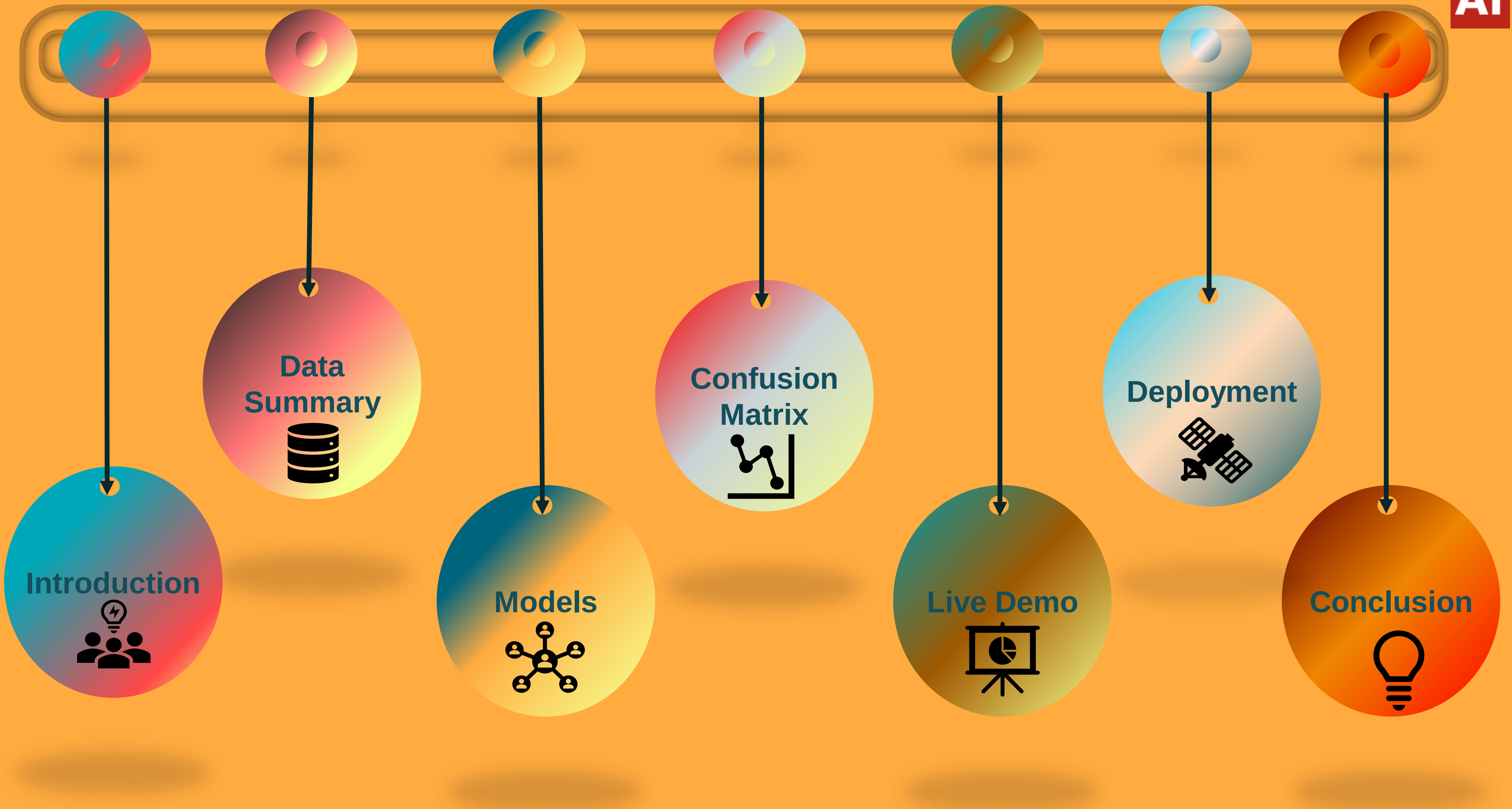
# “Face Emotion Recognition”

-By “Shailendra Dubey”

*Before Diving deep into the Presentation, let’s first see*

# Agenda

*Of the Presentation.*



Introduction

Data Summary

Models

Confusion Matrix

Live Demo

Deployment

Conclusion

***Let’s start with***

# Introduction

***Emotions are an inevitable part of personal communication. It can be expressed in many different forms, which may or may not be observed with the naked eye. Therefore, with the right tools, any indications preceding or following them can be subject to detection and recognition. There has been an increase in the need to detect a person’s emotions in the past few years and increasing interest in human emotion recognition in various fields including, but not limited to, human-computer interfaces, animation, medicine, security, diagnostics for Autism Spectrum Disorders (ASD) in children, and urban sound perception. Emotion recognition can be performed using different features, such as facial expressions, speech, EEG, and even text. Among these features, facial expressions are one of the most popular, if not the most popular, due to a number of reasons; they are visible, they contain many useful features for emotion recognition, and it is easier to collect a large dataset of faces (than other means for human recognition).***

# Why Face Emotion Recognition ?

The Indian education landscape has been undergoing rapid changes for the past 10 years owing to the advancement of web based learning services, specifically, eLearning platforms. Global E-learning is estimated to witness an 8X over the next 5 years to reach USD 2B in 2021. India is expected to grow with a CAGR of 44% crossing the 10M users mark in 2021. Although the market is growing on a rapid scale, there are major challenges associated with digital learning when compared with brick and mortar classrooms. One of many challenges is how to ensure quality learning for students. Digital platforms might overpower physical classrooms in terms of content quality but when it comes to understanding whether students are able to grasp the content in a live class scenario is yet an open-end challenge. In a physical classroom during a lecturing teacher can see the faces and assess the emotion of the class and tune their lecture accordingly, whether he is going fast or slow. He can identify students who need special attention. Digital classrooms are conducted via video telephony software program (ex-Zoom) where it’s not possible for medium scale class (25-50) to see all students and access the mood. Because of this drawback, students are not focusing on content due to lack of surveillance. While digital platforms have limitations in terms of physical surveillance but it comes with the power of data and machines which can work for you. It provides data in the form of video, audio, and texts which can be analyzed using deep learning algorithms. Deep learning backed system not only solves the surveillance issue, but it also removes the human bias from the system, and all information is no longer in the teacher’s brain rather translated in numbers that can be analyzed and tracked. I will solve the above-mentioned challenge by applying deep learning algorithms to live video data. The solution to this problem is by recognizing facial emotions.

*Let's now proceed for*

AI

# Data Summary

**There are many datasets on Kaggle to work on face emotion recognition project but the dataset that we have used can be downloaded from the link below-** https://www.kaggle.com/datasets/msambare/fer2013

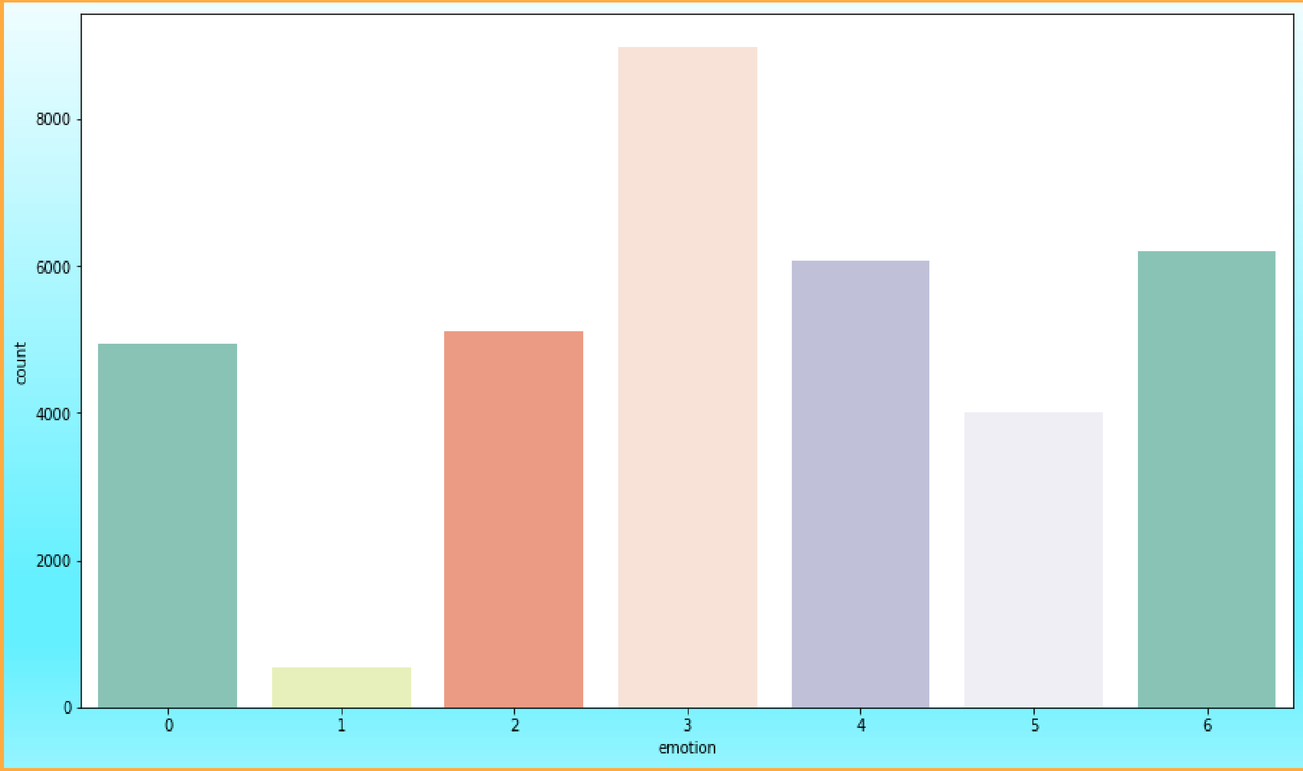
**This dataset consists of 35887 grayscale, 48x48 sized face images with seven emotions - angry, disgusted, fearful, happy, neutral, sad and surprised.**

***Our Dataset consists of different labels such as 0,1,2,3,4,5,and 6.***

| Labels | Emotions | Emojis | Images for Training | Images for testing |
|---|---|---|---|---|
| 0 | Anger | 😠 | 3995 | 958 |
| 1 | Disgust | 😟 | 436 | 111 |
| 2 | Fear | 😮 | 4097 | 1024 |
| 3 | happy | 🙂 | 7215 | 1774 |
| 4 | Sad | ☹ | 4830 | 1247 |
| 5 | Surprise | 🙄 | 3171 | 831 |
| 6 | Neutral | 😑 | 4965 | 1233 |

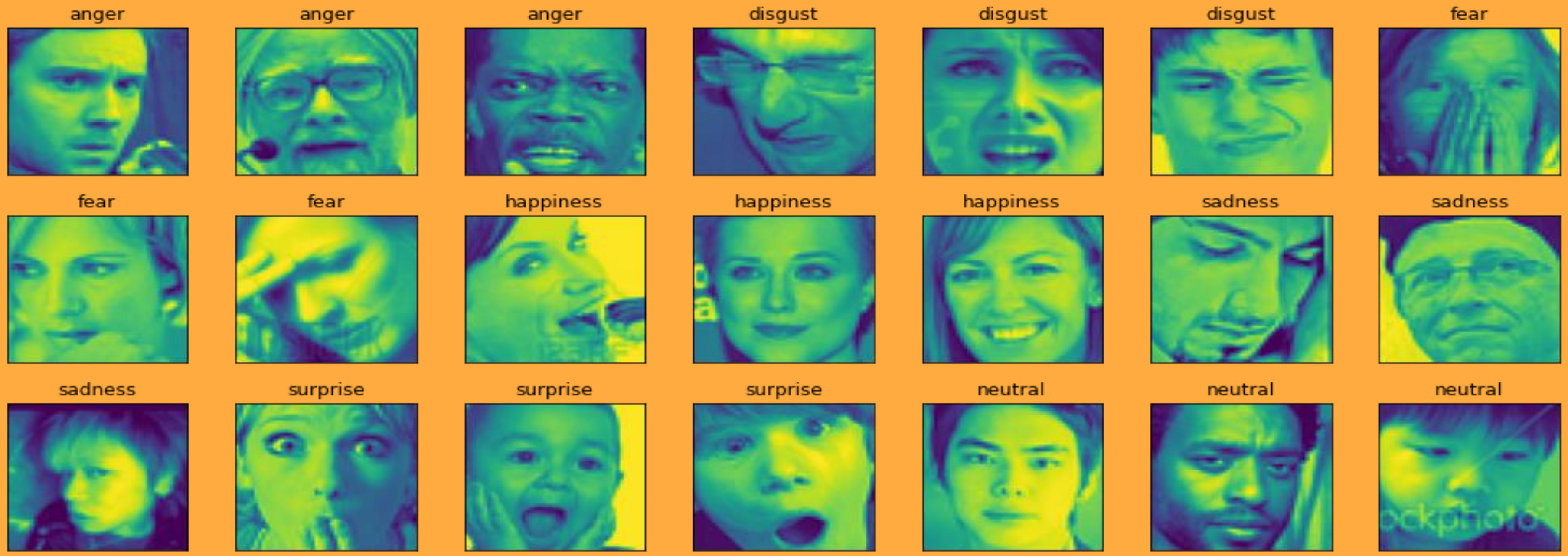
# Data Summary

- After looking at the bar plot, we can see that the bulk of the classes are **3:Happy**, **4:Sad**, and **6:Neutral**, while **1:Disgustand 5:Surprise** are less common, and **0:Anger** is also common.



# Data Summary

***We can see the different images with different emotions present in our Dataset-***



*Now let’s proceed for the different models that we are going to use.*

# Models

# Different Models

1 Deepface

2 Resnet50

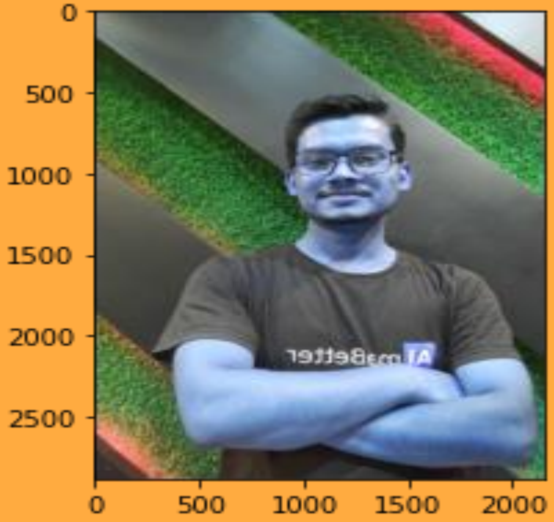
3 Xception

4 Custom CNN

*Let’s see our first  Model*

# Deepface

**Deepface is a lightweight face recognition and facial analysis attribute framework  for python**.



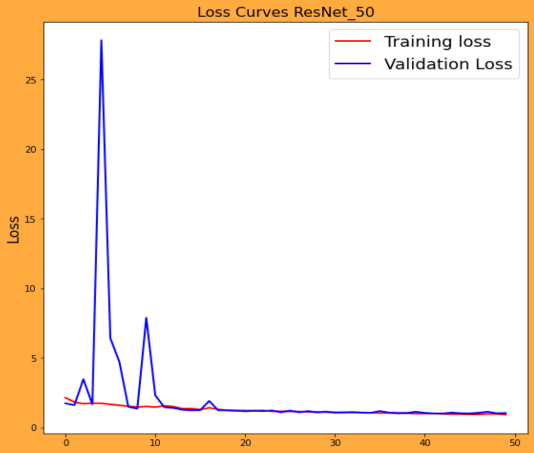
**I imported my above image in which I am happy but deepface model gives us** “`31 years old latino hispanic neutral Man`”. **To get better result, we decided to train our own model**.

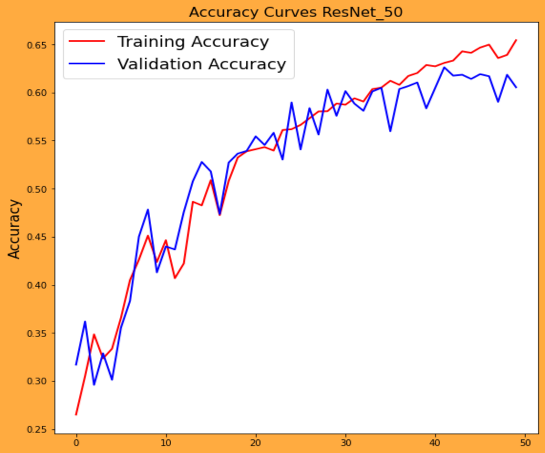
*Let’s see our second model*

# Resnet 50

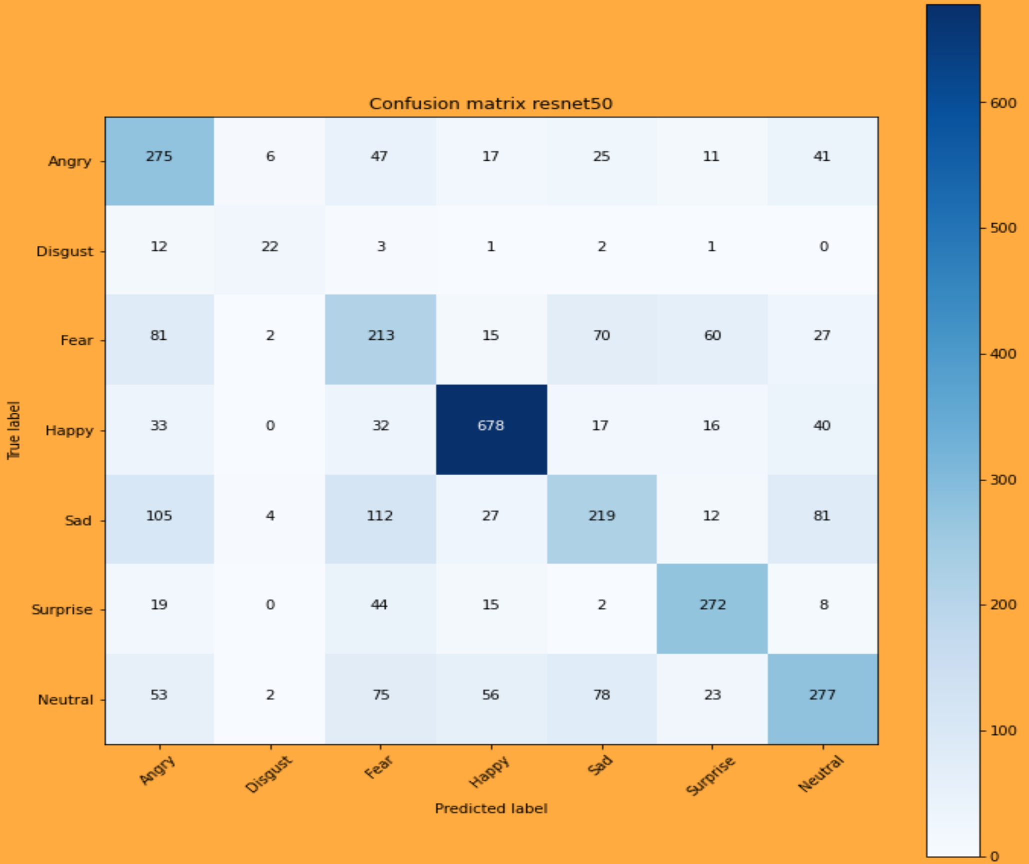
ResNet50 is the first pre-trained model we explored. ResNet50 is a deep residual network with 50 layers. It is defined in Keras with 175 layers. We replaced the original output layer with one FC layer of size 1000 and a softmax output layer of 7 emotion classes. We used Adam as our optimizer after training for 50 epochs using Adam and a batch size of 785, we achieved 60.54% accuracy on the test set and 65.43% on the train set. There is much less over-fitting.

## Here, we can see loss and accuracy curve of resnet 50.

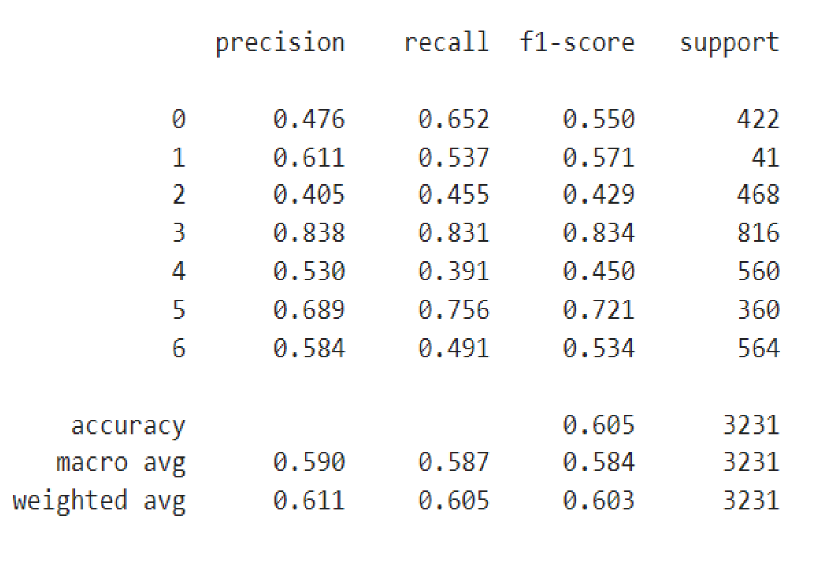

Loss Curves ResNet_50


Accuracy Curves ResNet_50

## *Let’s see the confusion matrix and classification report of resnet 50.*



| | Angry | Disgust | Fear | Happy | Sad | Surprise | Neutral |
|---|---|---|---|---|---|---|---|
| Angry | 275 | 6 | 47 | 17 | 25 | 11 | 41 |
| Disgust | 12 | 22 | 3 | 1 | 2 | 1 | 0 |
| Fear | 81 | 2 | 213 | 15 | 70 | 60 | 27 |
| Happy | 33 | 0 | 32 | 678 | 17 | 16 | 40 |
| Sad | 105 | 4 | 112 | 27 | 219 | 12 | 81 |
| Surprise | 19 | 0 | 44 | 15 | 2 | 272 | 8 |
| Neutral | 53 | 2 | 75 | 56 | 78 | 23 | 277 |



```
              precision    recall  f1-score   support

           0      0.476     0.652     0.550       422
           1      0.611     0.537     0.571        41
           2      0.405     0.455     0.429       468
           3      0.838     0.831     0.834       816
           4      0.530     0.391     0.450       560
           5      0.689     0.756     0.721       360
           6      0.584     0.491     0.534       564

    accuracy                          0.605      3231
   macro avg      0.590     0.587     0.584      3231
weighted avg      0.611     0.605     0.603      3231
```

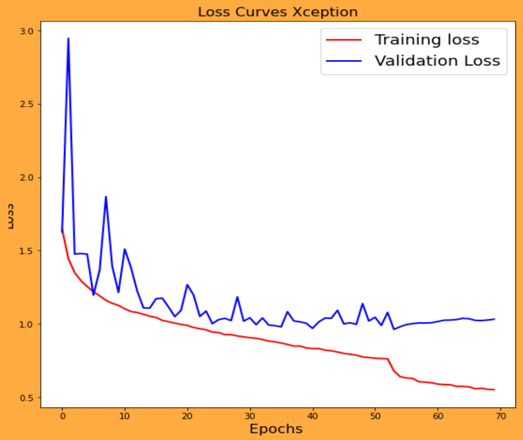
**From the confusion metrix above we observe the emotions that are correctly predicted w.r.t actual emotions.**

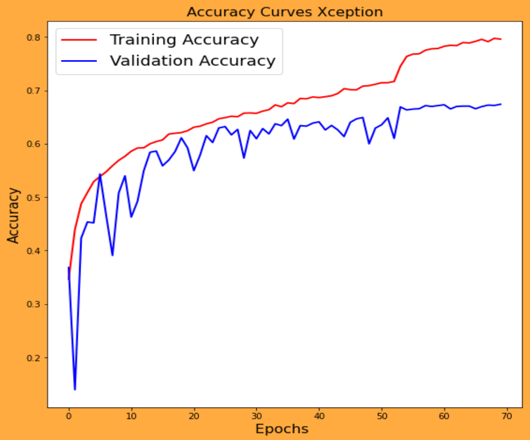
*Let’s now proceed for our third model.*

# Xception

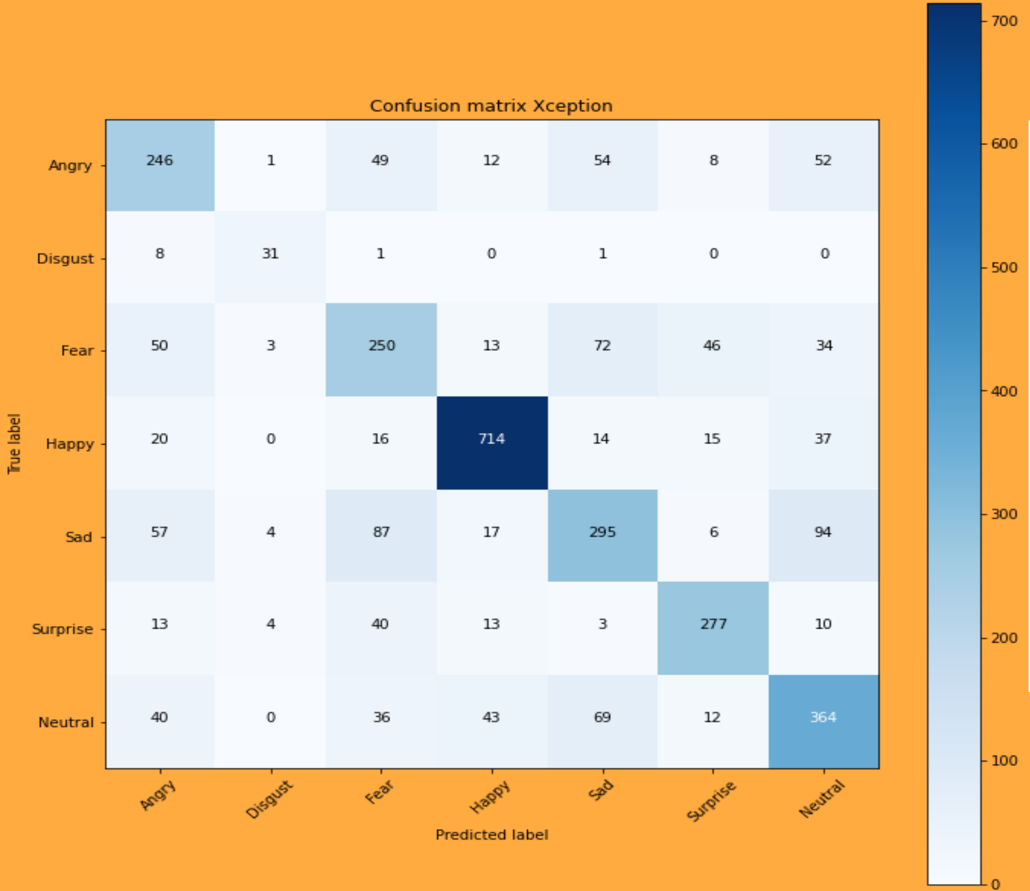
**Xception architecture is a linear stack of depth wise separable convolution layers with residual connections. This makes the architecture very easy to define and modify; it takes only 30 to 40 lines of code using a high level library such as Keras or Tensorflow not unlike an architecture such as VGG-16, but rather un- like architectures such as Inception V2 or V3 which are far more complex to define. We used Adam as our optimizer after training for 70 epochs using Adam and a batch size of 785, we achieved 67% accuracy on the test set and 79% on training set.**

## Here, we can see loss and accuracy curve of Xception.



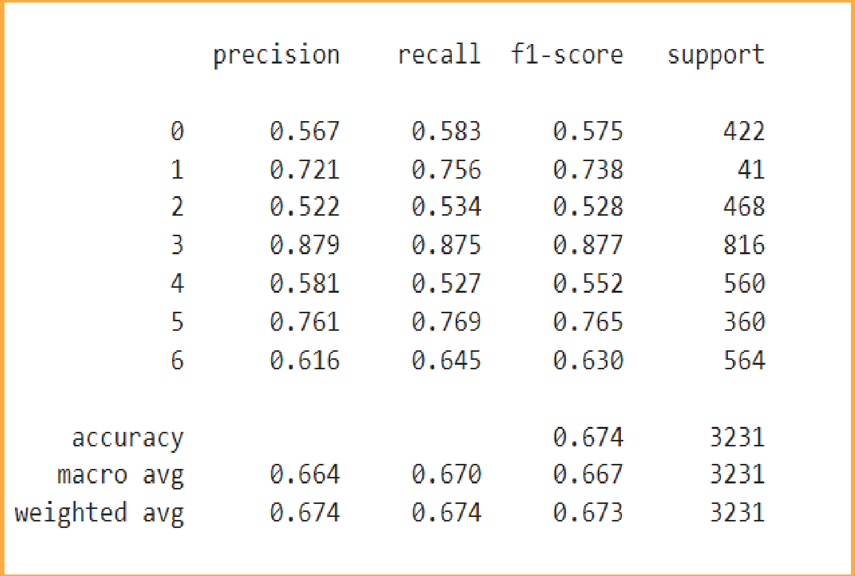


## *Let’s see the confusion matrix and classification report of xception.*



Confusion matrix Xception

| True label \ Predicted label | Angry | Disgust | Fear | Happy | Sad | Surprise | Neutral |
|---|---|---|---|---|---|---|---|
| Angry | 246 | 1 | 49 | 12 | 54 | 8 | 52 |
| Disgust | 8 | 31 | 1 | 0 | 1 | 0 | 0 |
| Fear | 50 | 3 | 250 | 13 | 72 | 46 | 34 |
| Happy | 20 | 0 | 16 | 714 | 14 | 15 | 37 |
| Sad | 57 | 4 | 87 | 17 | 295 | 6 | 94 |
| Surprise | 13 | 4 | 40 | 13 | 3 | 277 | 10 |
| Neutral | 40 | 0 | 36 | 43 | 69 | 12 | 364 |



```
              precision    recall  f1-score   support

           0      0.567     0.583     0.575       422
           1      0.721     0.756     0.738        41
           2      0.522     0.534     0.528       468
           3      0.879     0.875     0.877       816
           4      0.581     0.527     0.552       560
           5      0.761     0.769     0.765       360
           6      0.616     0.645     0.630       564

    accuracy                          0.674      3231
   macro avg      0.664     0.670     0.667      3231
weighted avg      0.674     0.674     0.673      3231
```

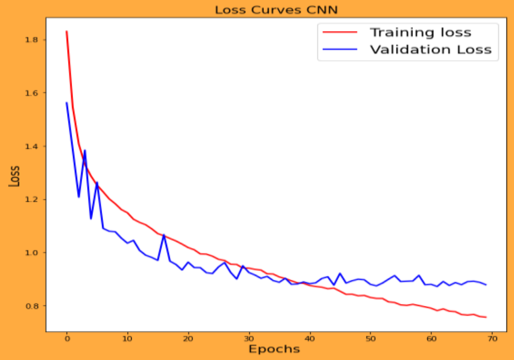
**From the confusion metrix above we observe the emotions that are correctly predicted w.r.t actual emotions.**

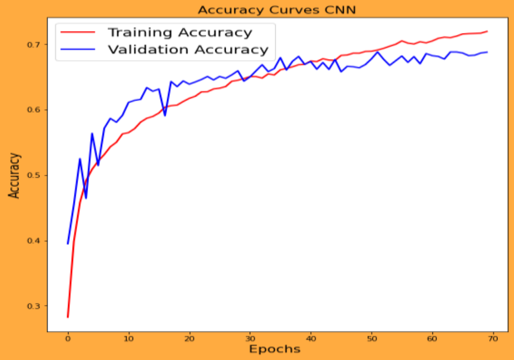
*Let’s now proceed for our fourth model.*

# Custom CNN

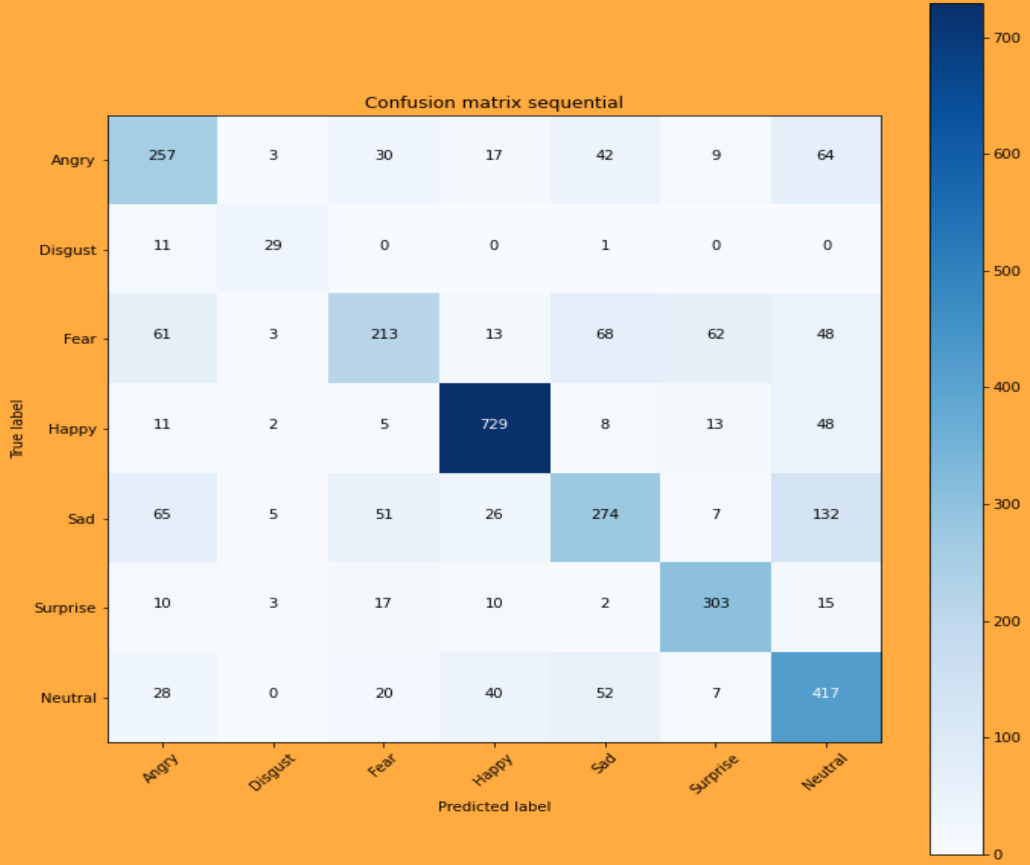
A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. We used Adam as our optimizer after training for 70 epochs using Adam with minimum learning rate 0.00001 and a batch size of 785, we achieved 68 % accuracy on the test set and 71% as train accuracy. One drawback of the system is the some Disgust faces are showing Neutral .Because less no. of disgust faces are given to train .This may be the reason. I thought it was a good score should improve the score. Thus I decided that I will deploy the model.

## Here, we can see loss and accuracy curve of CNN.

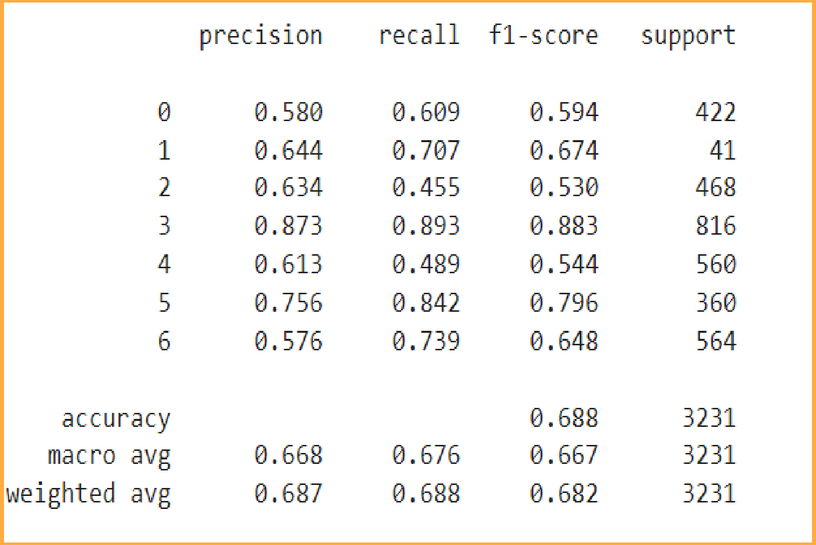




## *Let’s see the confusion matrix and classification report of custom cnn.*



| True label \ Predicted label | Angry | Disgust | Fear | Happy | Sad | Surprise | Neutral |
|---|---|---|---|---|---|---|---|
| Angry | 257 | 3 | 30 | 17 | 42 | 9 | 64 |
| Disgust | 11 | 29 | 0 | 0 | 1 | 0 | 0 |
| Fear | 61 | 3 | 213 | 13 | 68 | 62 | 48 |
| Happy | 11 | 2 | 5 | 729 | 8 | 13 | 48 |
| Sad | 65 | 5 | 51 | 26 | 274 | 7 | 132 |
| Surprise | 10 | 3 | 17 | 10 | 2 | 303 | 15 |
| Neutral | 28 | 0 | 20 | 40 | 52 | 7 | 417 |



```
              precision    recall  f1-score   support

           0      0.580     0.609     0.594       422
           1      0.644     0.707     0.674        41
           2      0.634     0.455     0.530       468
           3      0.873     0.893     0.883       816
           4      0.613     0.489     0.544       560
           5      0.756     0.842     0.796       360
           6      0.576     0.739     0.648       564

    accuracy                          0.688      3231
   macro avg      0.668     0.676     0.667      3231
weighted avg      0.687     0.688     0.682      3231
```

**From the confusion metrix above we observe the emotions that are correctly predicted w.r.t actual emotions**

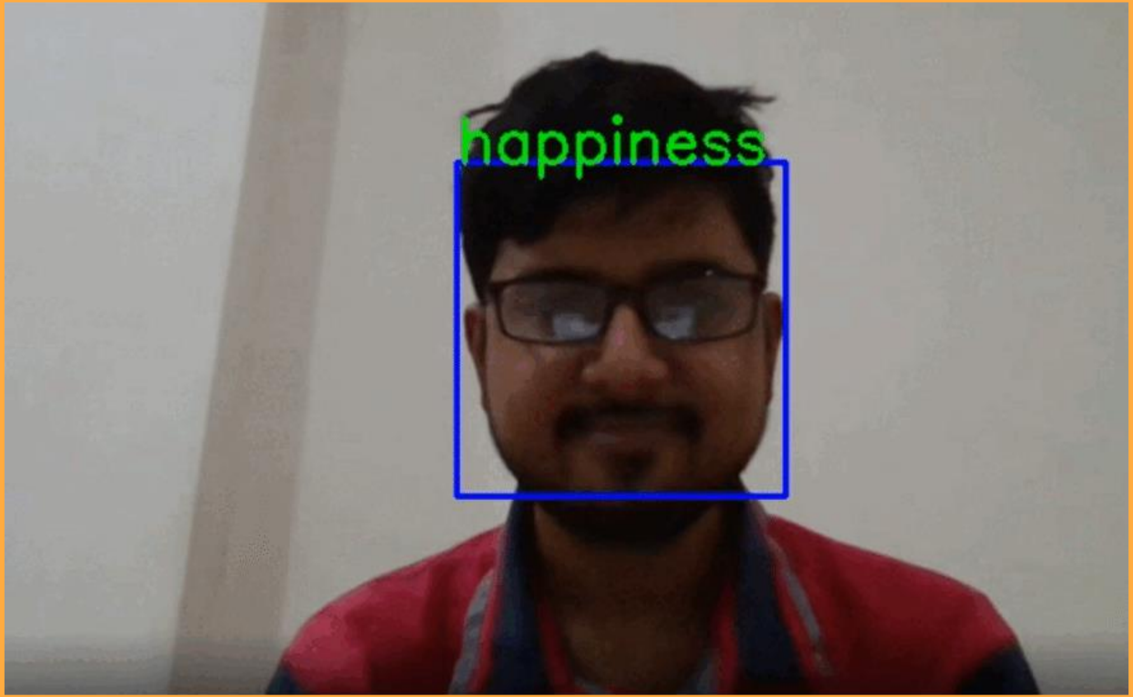
# Live Demo

- We created patterns for detecting and predicting single faces and as well as multiple faces using OpenCV video capture in local webcam.

- For Web-app, OpenCV can’t be used. Thus, using Streamlit-Webrtc for front-End application development.

- I have also included the video of my WebApp working in Local host.

# Live Demo



*Let’s now deploy our model*.

# Deployment

- We deployed the app in Heroku if you saw in the starting section of GitHub repo you see the all the requirement files are there for creating an app on Heroku.
- But due to high slug size the buffering takes time so we have ran our app working on local and it ran properly and app is also fine also we’ve included video on GitHub repo.
- App Link--

# Conclusion

- We used different models but the Custom CNN model works pretty fine because it gives good accuracy result on test dataset.

- We build the WebApp using Streamlit and deployed in Heroku.

- The model which was created by custom CNN model gave training accuracy of  71% and test accuracy of 68%.

- I have also included the video of my WebApp working in Local.

# Challenges

- ✓ To handle a large image dataset.
- ✓ Selecting the number of filters and neurons.
- ✓ Select a batch size to avoid the system crashing.
- ✓ GPU connection to our notebook.
- ✓ Deployment (Biggest Problem).

# Thank You