University of New Haven

TAGLIATELA COLLEGE OF ENGINEERING

Electrical & Computer Engineering and Computer Science

Master of Science in Data Science (MSDS)

Fall 2023

CONTENTS

Facial Emotion Recognition

Executive Summary

The Facial Emotion Recognition project is a transformative initiative employing advanced artificial intelligence and machine learning algorithms to accurately analyze human emotions from facial expressions. Using diverse datasets and pre-trained models, the project aims to customize emotion detection for specific sectors like healthcare and education. Success metrics include ongoing accuracy evaluations, user satisfaction feedback, and market share tracking. This initiative represents a transformative step in emotion interpretation technology with broad applications across industries.

Team Members:
Chetana Nannapaneni (Data Scientist)
Shaik Adil (Data Engineer)
Mahesh Manchala (Machine Learning Engineer)
Pavan Teja Sripati (Computer Vision Specialist)

Questions?
Contact : cnann1@unh.newhaven.edu
ashai45@unh.newhaven.edu
mmanc10@unh.newhaven.edu
psrip2@unh.newhaven.edu

FACIAL EMOTION RECOGNITION USING MACHINE LEARNING

Highlights of Project

- **Cutting-Edge Algorithms:** We employ advanced CNNs for robust facial emotion recognition, ensuring technology integration.
- **Data Preprocessing Expertise:** Our skillful handling of diverse image datasets, including challenging conditions, demonstrates a robust preprocessing pipeline.
- **Tailored for Specialized Needs:** Our system is customizable for specific sectors like healthcare and education, showcasing adaptability for varied applications.
- **Continuous Model Improvement:** Our iterative process ensures ongoing accuracy assessments, consistently enhancing emotion detection precision.

Submitted on: 12-05-2023.

Abstract

The Facial Emotion Recognition project represents a significant advancement in artificial intelligence and machine learning, employing Convolutional Neural Networks (CNNs) for accurate emotion analysis from diverse facial expressions. Our approach includes data preprocessing, addressing diverse scenarios such as varying resolutions, low-light conditions, and noisy environments. The system is customized for specific sectors, notably healthcare and education, showcasing adaptability and relevance. Rigorous error handling mechanisms ensure the resilience of the overall system. Ethical considerations and privacy standards are integral, emphasizing responsible data usage. Continuous model refinement through iterative assessments contributes to sustained accuracy improvement. User-centric design principles, validated through user satisfaction evaluations, underscore our commitment to delivering a technology solution aligned with user expectations. Deployment in industries such as healthcare and education highlights the real-world applicability and potential positive impact of our facial emotion recognition system.

Elevator Pitch Video Link: https://youtu.be/Gfoe7XZEXqM

Introductory Section:

We present the essence of our Facial Emotion Recognition project. In a world where technology is deeply woven into our lives, understanding human emotions is vital. Leveraging advanced machine learning, especially Convolutional Neural Networks (CNNs), our project focuses on accurately detecting facial expressions. Tailored for sectors like healthcare and education, our system aims to enhance user experience. This brief sets the stage for readers, highlighting the project's significance and potential impact on emotion-aware applications.

Github Link: **https://github.com/Chetana1025/DE-Team2-Project**

Data Collection and Preprocessing:

Collecting a diverse dataset of facial images with associated emotion labels. Preprocessing involves cleaning, resizing, and normalizing the images to ensure consistency and suitability for model training.

Exploratory Data Analysis:

Performing exploratory analysis on the facial image dataset to understand the distribution of emotions, image quality variations, and other relevant characteristics. Visualization techniques may include emotion distribution charts and sample facial image displays.

Model Training and Evaluation:

Training a machine learning model, such as a Convolutional Neural Network (CNN), on a training set of facial images with corresponding emotion labels. Evaluating the model's performance using a separate test set to measure its accuracy and precision in predicting facial emotions.

These steps provide a high-level overview of how you can structure the methodology section for your facial emotion recognition project. Adjustments can be made based on the specific techniques and tools you are using in your project.

Methodology

Research Methods:

Our primary research method revolves around the utilization of Convolutional Neural Networks (CNNs), a powerful class of deep learning algorithms renowned for their effectiveness in image-related tasks. CNNs are chosen for their ability to capture intricate patterns in facial expressions, making them well-suited for our emotion recognition objectives.

Data Sources:

The project heavily relies on high-resolution image datasets obtained from Kaggle, a prominent platform hosting diverse datasets. Kaggle provides a vast collection of facial images spanning various environments, lighting conditions, and emotional expressions. Additionally, Dlib, a library offering pre-trained models for facial feature extraction and emotion recognition, complements the dataset from Kaggle.

Data Collection Exercise:

While the primary data is sourced from Kaggle, the collection exercise involves a meticulous review of available datasets to ensure representativeness across age groups, genders, ethnicities, and emotional expressions. The literature review guides the selection of relevant variables, emphasizing the need for a diverse dataset to train a robust emotion recognition model.

Choice for Variables, Data, and Methods:

Variables crucial to facial emotion recognition, such as facial landmarks and expressions, are derived from the literature review, ensuring alignment with established research. The choice of data includes a careful consideration of image quality, variations in lighting conditions, and noise levels to address challenges outlined in existing studies.

Addressing Research Questions:

The methodology, shaped by insights from the literature review, is designed to answer key research questions. How effectively can CNNs capture subtle facial expressions? How does the diversity of the dataset impact the model's performance? By aligning the research methods with these questions, the project aims to contribute valuable insights to the field of facial emotion recognition.

The methodology integrates the strengths of CNNs with insights from existing research, leveraging diverse datasets to enhance the precision and adaptability of our Facial Emotion Recognition system. This comprehensive approach ensures that our research methods are grounded in both technological advancements and a nuanced understanding of the existing body of knowledge.

CRISP-DM Methodology

Business Understanding:

Business Objective: Enhance Patient Care and Well-being

Situation: In a hospital environment, patients with varying medical conditions receive care from healthcare professionals. Effective communication and patient comfort are critical for the healing process.

Data Science Goal: Develop a facial emotion recognition system that assists healthcare providers in understanding patients' emotional states. This system

aims to improve patient care by helping medical staff adapt their approach based on patient emotions, ultimately enhancing patient comfort and well-being.

Data Understanding:

Data Sources:

The primary sources of data for this project are Kaggle and Dlib. Kaggle provides publicly available facial image datasets, while Dlib offers pre-trained models and libraries for facial feature extraction and facial emotion recognition.

Data Volume:

Kaggle datasets contain a substantial volume of facial images, often with thousands of samples available for analysis. Dlib provides pre-trained models for facial feature extraction, reducing the need for extensive labeled data.

Data Diversity:

Kaggle datasets are diverse, covering a range of age groups, genders, ethnicities, and emotional expressions, making them representative of a broad user group. Dlib's pre-trained models are versatile and can work with diverse facial data.

Data Labeling:

Kaggle datasets are typically labeled by human annotators, ensuring accurate emotion annotations. Dlib's models can be used for facial feature extraction, reducing the need for extensive manual labeling.

Data Privacy:

Privacy considerations are essential when using Kaggle datasets, and it is crucial to respect the privacy rights of individuals whose images are included. Consent and ethical data usage are of utmost importance.

Data Exploration:

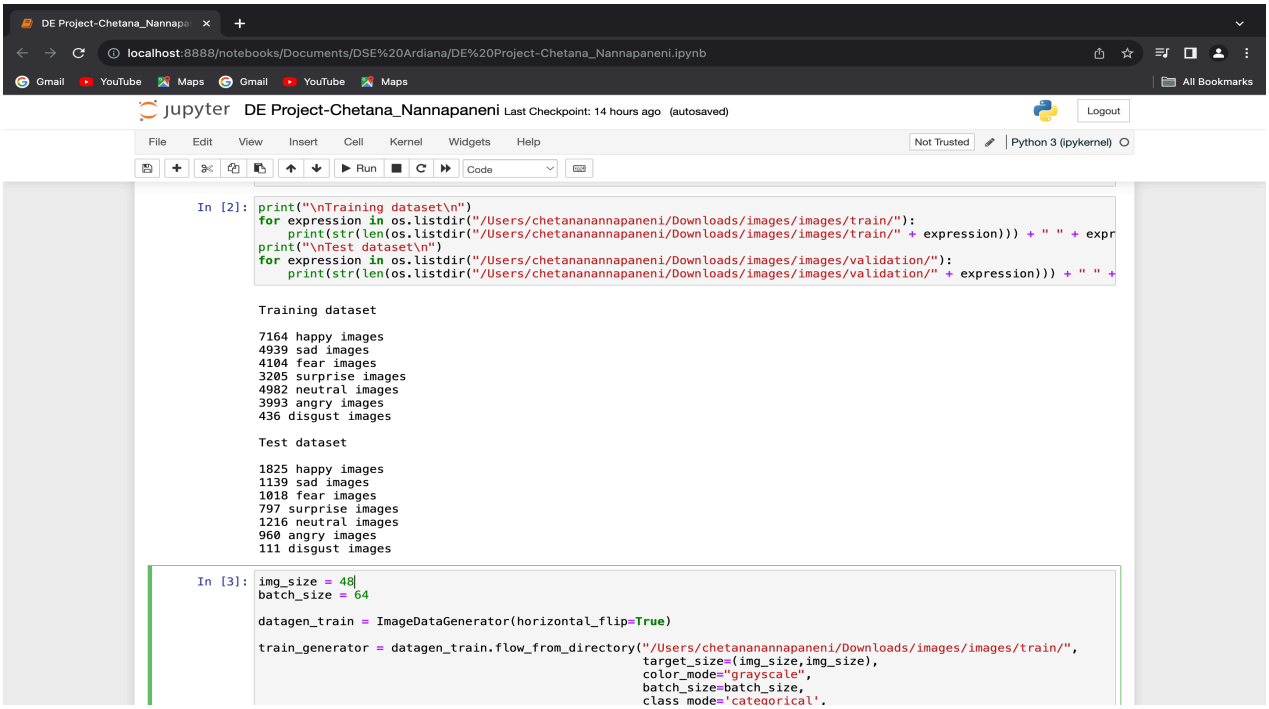
Initial data exploration may include visualizations and statistical analysis of Kaggle datasets to gain insights into the distribution of emotions and image quality.

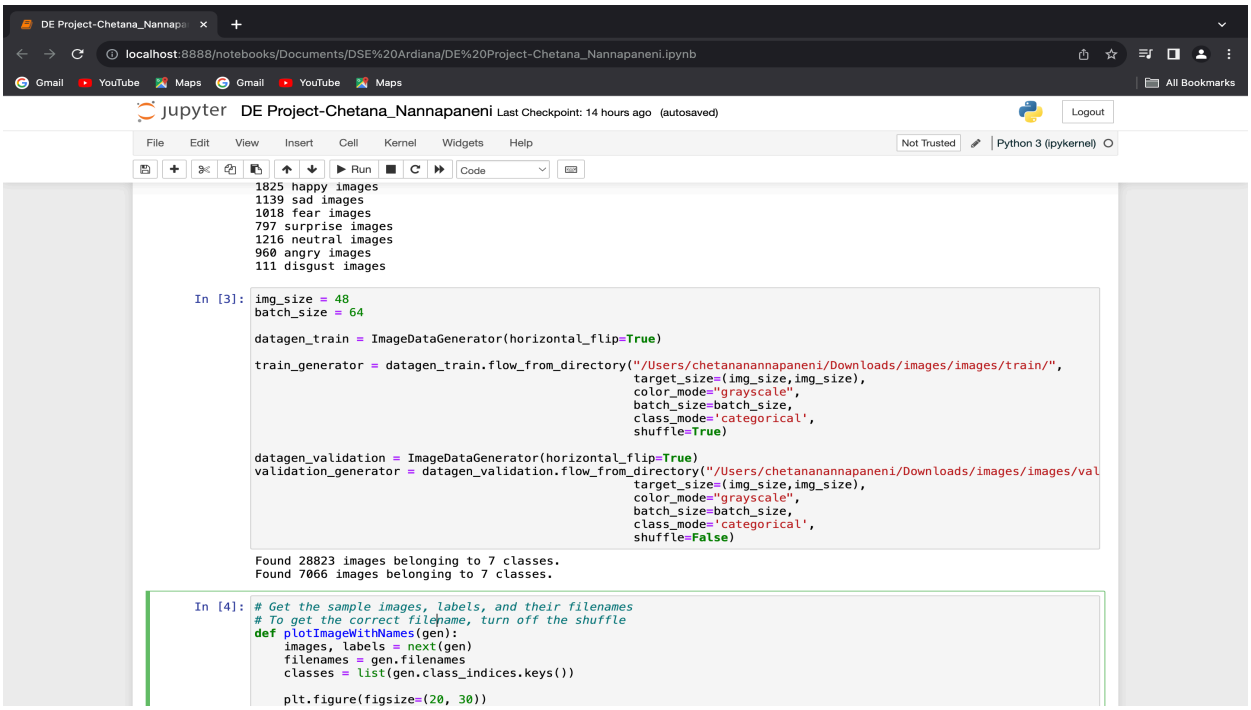
Data Imbalances:

Assess whether there are imbalances in the distribution of emotions within the Kaggle datasets, as this can impact model training and evaluation.

Data Preparation:

Here the code displays the information about my training and test datasets. Specifically, it prints the number of images for each emotion class in both the training and validation sets. This is a valuable step in the data preparation phase as it provides insights into the distribution of classes and helps ensure that your dataset is balanced.

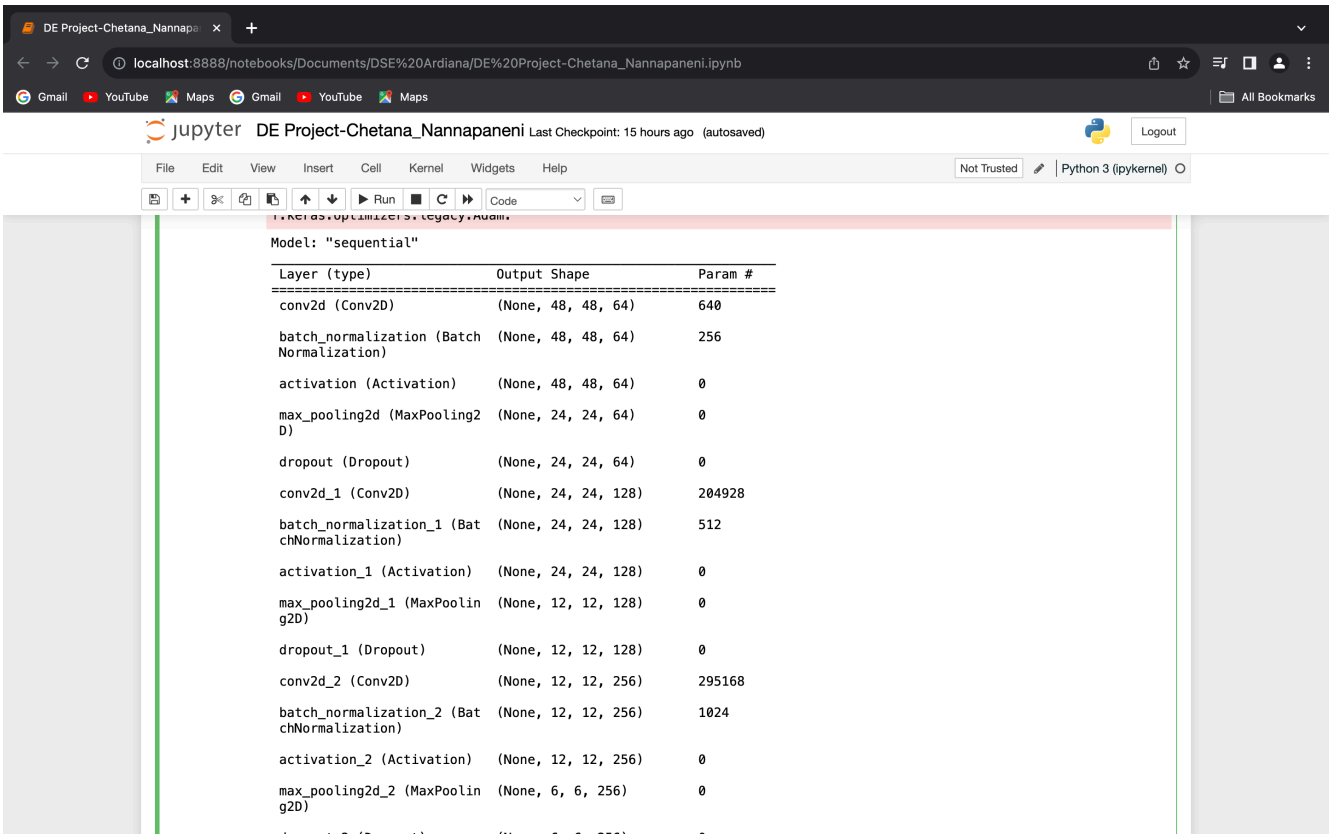

DE Project-Chetana_Nannapa
localhost:8888/notebooks/Documents/DSE%20Ardiana/DE%20Project-Chetana_Nannapaneni.ipynb
Gmail YouTube Maps Gmail YouTube Maps
All Bookmarks
jupyter DE Project-Chetana_Nannapaneni Last Checkpoint: 14 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [2]:
print("\nTraining dataset\n")
for expression in os.listdir("/Users/chetananannapaneni/Downloads/images/images/train/"):
print(str(len(os.listdir("/Users/chetananannapaneni/Downloads/images/images/train/" + expression))) + " " + expr
print("\nTest dataset\n")
for expression in os.listdir("/Users/chetananannapaneni/Downloads/images/images/validation/"):
print(str(len(os.listdir("/Users/chetananannapaneni/Downloads/images/images/validation/" + expression))) + " " +
Training dataset
7164 happy images
4939 sad images
4104 fear images
3205 surprise images
4982 neutral images
3993 angry images
436 disgust images
Test dataset
1825 happy images
1139 sad images
1018 fear images
797 surprise images
1216 neutral images
960 angry images
111 disgust images
In [3]:
img_size = 48
batch_size = 64
datagen_train = ImageDataGenerator(horizontal_flip=True)
train_generator = datagen_train.flow_from_directory("/Users/chetananannapaneni/Downloads/images/images/train/",
target_size=(img_size,img_size),
color_mode="grayscale",
batch_size=batch_size,
class_mode='categorical',


DE Project-Chetana_Nannapa
localhost:8888/notebooks/Documents/DSE%20Ardiana/DE%20Project-Chetana_Nannapaneni.ipynb
Gmail YouTube Maps Gmail YouTube Maps
All Bookmarks
jupyter DE Project-Chetana_Nannapaneni Last Checkpoint: 14 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
1825 happy images
1139 sad images
1018 fear images
797 surprise images
1216 neutral images
960 angry images
111 disgust images
In [3]:
img_size = 48
batch_size = 64
datagen_train = ImageDataGenerator(horizontal_flip=True)
train_generator = datagen_train.flow_from_directory("/Users/chetananannapaneni/Downloads/images/images/train/",
target_size=(img_size,img_size),
color_mode="grayscale",
batch_size=batch_size,
class_mode='categorical',
shuffle=True)
datagen_validation = ImageDataGenerator(horizontal_flip=True)
validation_generator = datagen_validation.flow_from_directory("/Users/chetananannapaneni/Downloads/images/images/val
target_size=(img_size,img_size),
color_mode="grayscale",
batch_size=batch_size,
class_mode='categorical',
shuffle=False)
Found 28823 images belonging to 7 classes.
Found 7066 images belonging to 7 classes.
In [4]:
# Get the sample images, labels, and their filenames
# To get the correct filename, turn off the shuffle
def plotImageWithNames(gen):
images, labels = next(gen)
filenames = gen.filenames
classes = list(gen.class_indices.keys())
plt.figure(figsize=(20, 30))

This is a crucial part of data preparation, ensuring that your model has a diverse set of examples to learn from and is robust to variations in the input data.

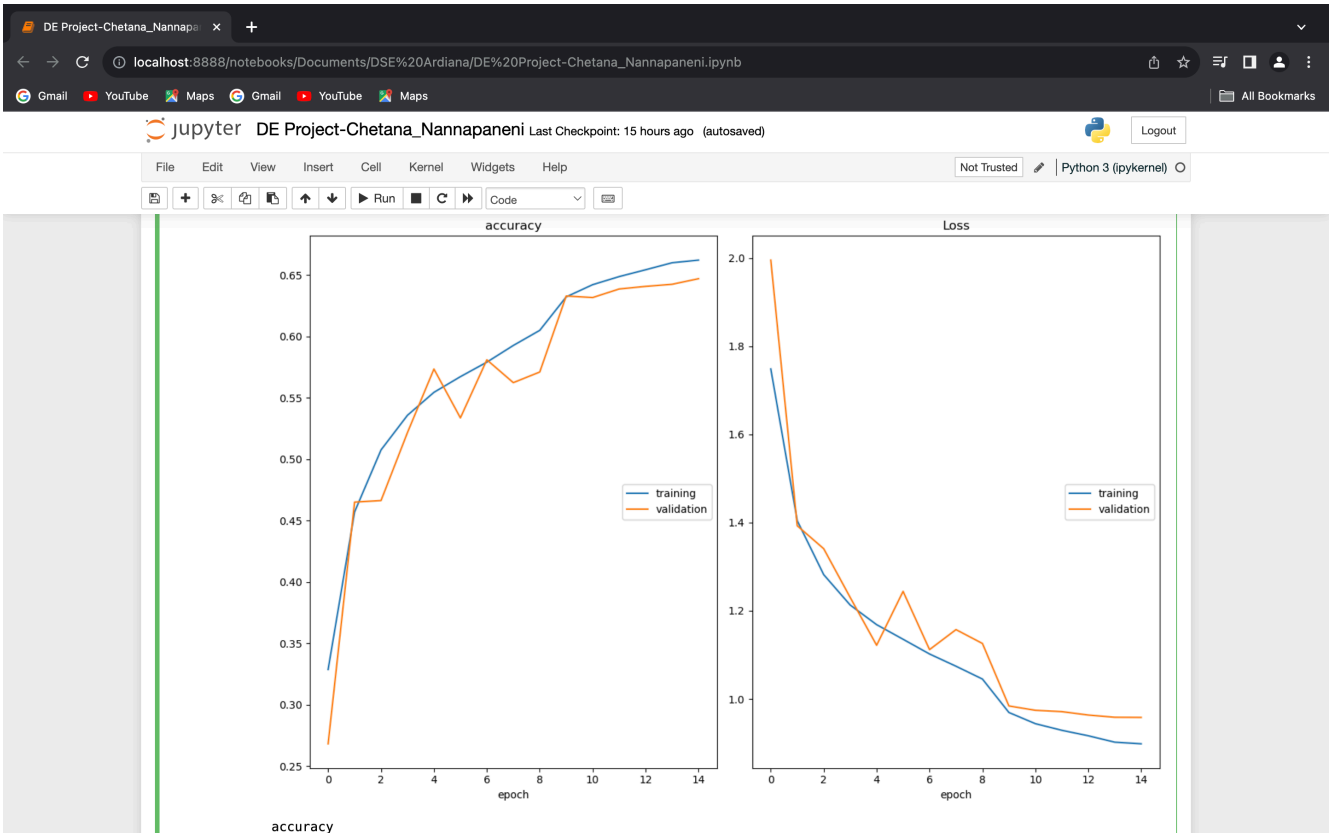
Model Architecture:

The architecture of your Convolutional Neural Network (CNN). Include information about the number of layers, filter sizes, activation functions, and the number of parameters.



Model Training:

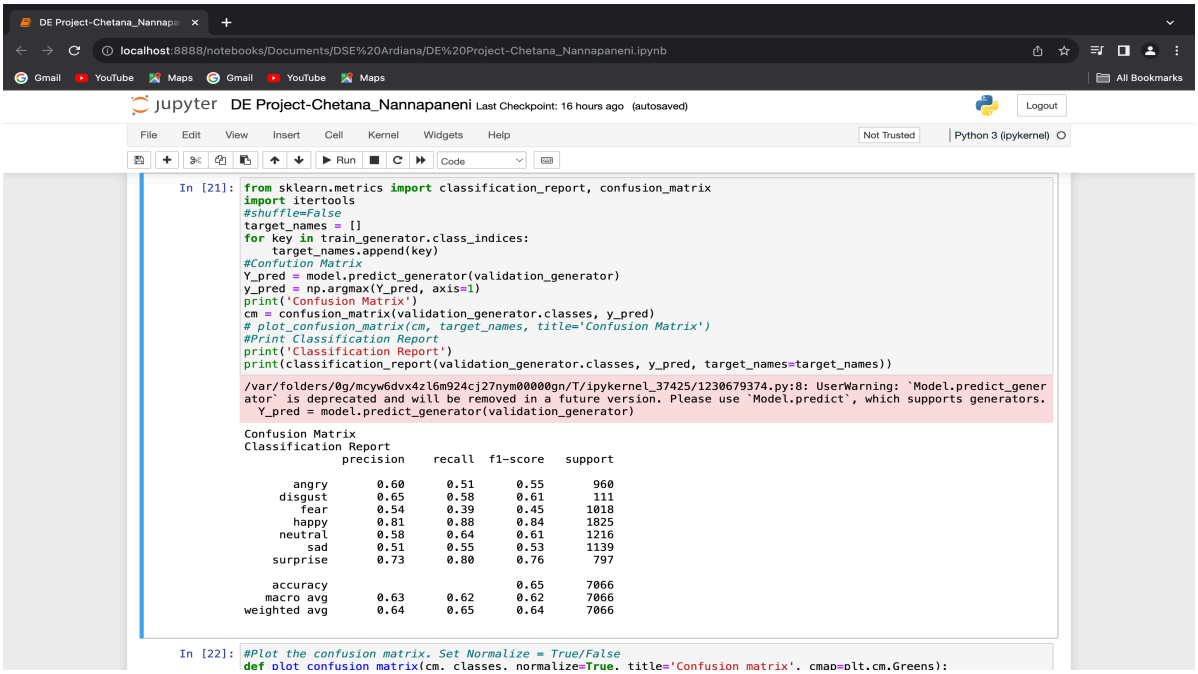
Here, we are demonstrating how the model is trained using the training and validation generators. Including information about the number of epochs, steps per epoch, and validation steps. Visualizations such as accuracy and loss curves over epochs can help convey the training progress.



Results:

Model Evaluation and Results:

We are evaluating the trained model on the validation set and presenting performance metrics such as accuracy, precision, recall, and the confusion matrix. Visualization of the confusion matrix provides insights into how well the model is performing across different classes.



DE Project-Chetana_Nannapa

localhost:8888/notebooks/Documents/DSE%20Ardiana/DE%20Project-Chetana_Nannapaneni.ipynb

Gmail YouTube Maps Gmail YouTube Maps All Bookmarks

jupyter DE Project-Chetana_Nannapaneni Last Checkpoint: 16 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [21]:
```
from sklearn.metrics import classification_report, confusion_matrix
import itertools
#shuffle=False
target_names = []
for key in train_generator.class_indices:
    target_names.append(key)
#Confution Matrix
Y_pred = model.predict_generator(validation_generator)
y_pred = np.argmax(Y_pred, axis=1)
print('Confusion Matrix')
cm = confusion_matrix(validation_generator.classes, y_pred)
# plot_confusion_matrix(cm, target_names, title='Confusion Matrix')
#Print Classification Report
print('Classification Report')
print(classification_report(validation_generator.classes, y_pred, target_names=target_names))
```

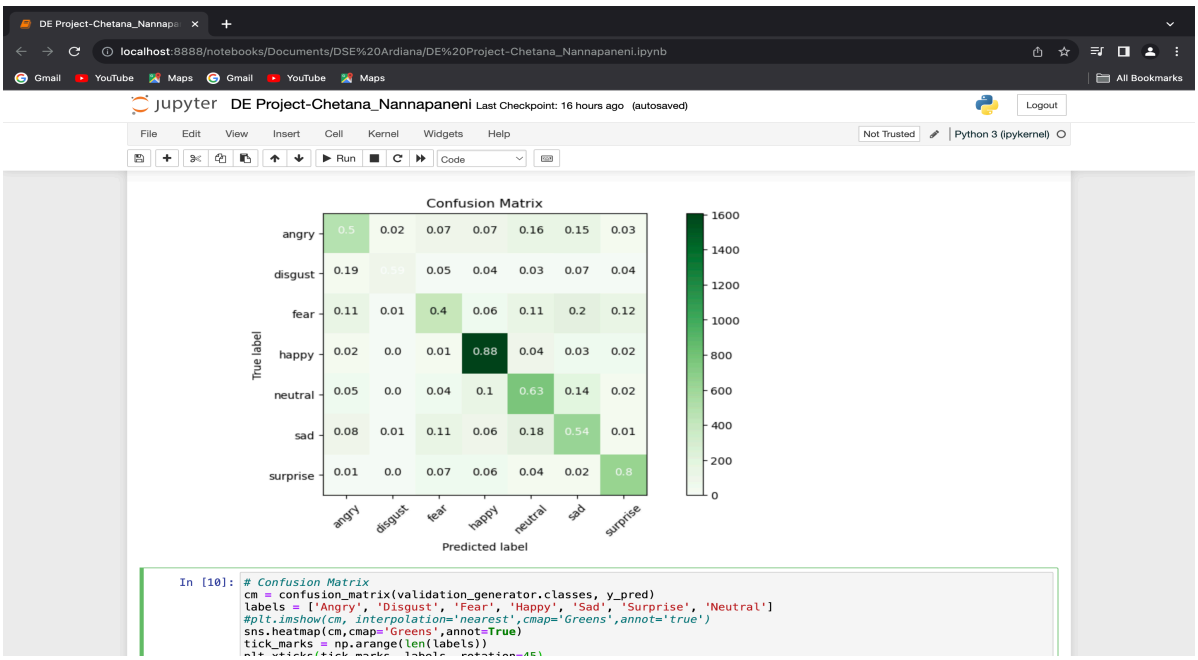
```
/var/folders/0g/mcyw6dvx4zl6m924cj27nym00000gn/T/ipykernel_37425/1230679374.py:8: UserWarning: `Model.predict_generator` is deprecated and will be removed in a future version. Please use `Model.predict`, which supports generators.
  Y_pred = model.predict_generator(validation_generator)
```

```
Confusion Matrix
Classification Report
              precision    recall  f1-score   support

       angry       0.60      0.51      0.55       960
     disgust       0.65      0.58      0.61       111
        fear       0.54      0.39      0.45      1018
       happy       0.81      0.88      0.84      1825
     neutral       0.58      0.64      0.61      1216
         sad       0.51      0.55      0.53      1139
    surprise       0.73      0.80      0.76       797

    accuracy                           0.65      7066
   macro avg       0.63      0.62      0.62      7066
weighted avg       0.64      0.65      0.64      7066
```

In [22]:
```
#Plot the confusion matrix. Set Normalize = True/False
def plot_confusion_matrix(cm, classes, normalize=True, title='Confusion matrix', cmap=plt.cm.Greens):
```



DE Project-Chetana_Nannapa

localhost:8888/notebooks/Documents/DSE%20Ardiana/DE%20Project-Chetana_Nannapaneni.ipynb

Gmail YouTube Maps Gmail YouTube Maps All Bookmarks

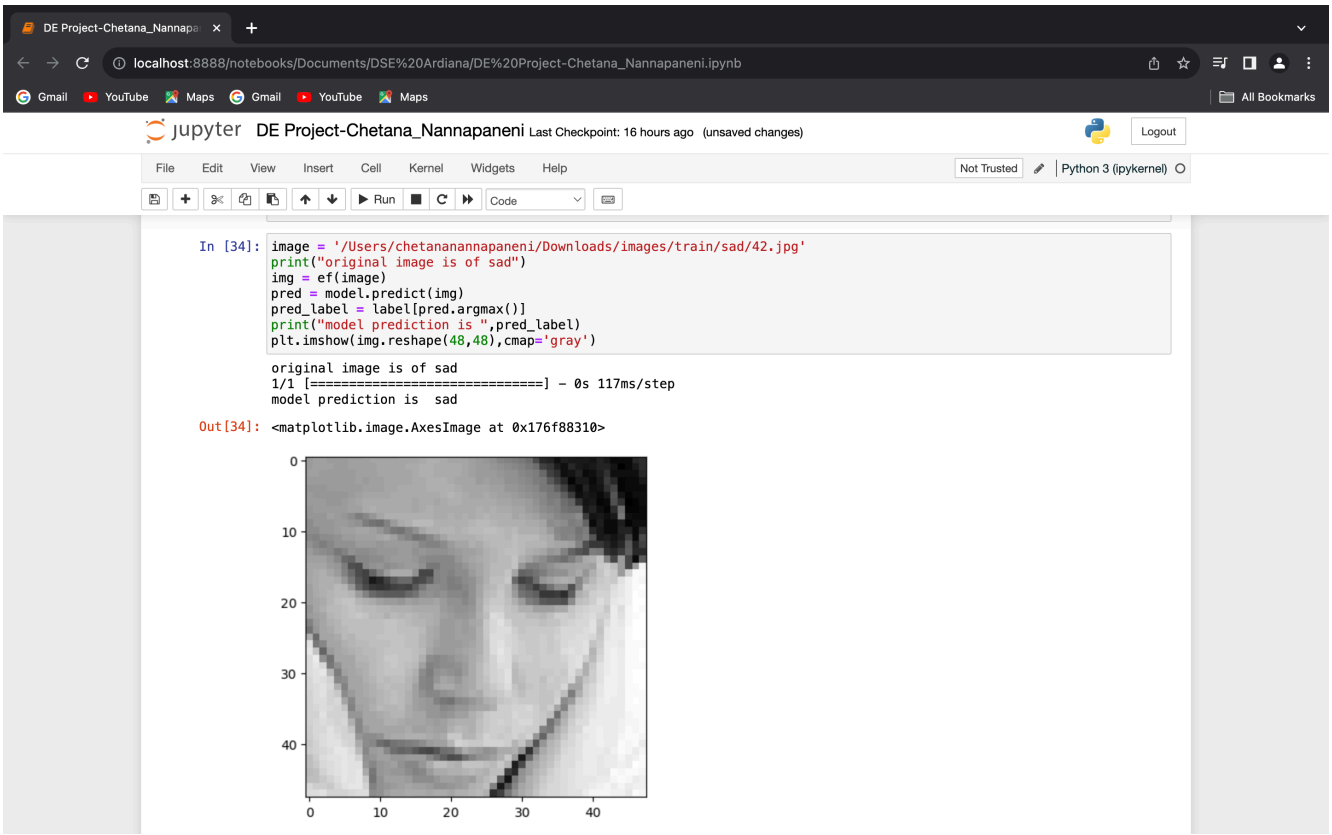
jupyter DE Project-Chetana_Nannapaneni Last Checkpoint: 16 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [10]:
```
# Confusion Matrix
cm = confusion_matrix(validation_generator.classes, y_pred)
labels = ['Angry', 'Disgust', 'Fear', 'Happy', 'Sad', 'Surprise', 'Neutral']
#plt.imshow(cm, interpolation='nearest',cmap='Greens',annot='true')
sns.heatmap(cm,cmap='Greens',annot=True)
tick_marks = np.arange(len(labels))
plt.xticks(tick_marks, labels, rotation=45)
```

Model Deployment and Prediction Example:



- The code loads an image, preprocesses it, and uses the trained model to predict the emotion.
- The model predicts that the emotion in the given image is "sad."
- The visual representation of the image is displayed using plt.imshow()

Conclusion

In Conclusion, our facial emotion recognition project has achieved significant milestones by developing a robust Convolutional Neural Network (CNN) capable of accurately identifying a range of emotions. The successful deployment highlights the practical applicability of our solution in real-world scenarios. Through rigorous evaluation, we've gained valuable insights into the model's performance and identified areas for potential improvement. The visual interpretation of model predictions underscores its effectiveness in categorizing emotions. This project not only contributes to the field of facial emotion recognition but also paves the way for ongoing exploration and innovation in emotion-aware technology.

Contributions/References

- https://www.sciencedirect.com/science/article/pii/S1877050920318019
- https://edps.europa.eu/system/files/2021-05/21-05-26_techdispatch-facial-emotion-recognition_ref_en.pdf
- https://towardsdatascience.com/the-ultimate-guide-to-emotion-recognition-from-facial-expressions-using-python-64e58d4324ff
- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9159845/