DermalScan:AI_Facial Skin Aging Detection App

Infosys SpringBoard Virtual Internship Program

Submitted By,

Priya Ghosal

Under the guidance of Mentor **Mr.Praveen**

Project Statement:

Facial skin aging shows signs like wrinkles, dark spots, and puffiness, which can be hard to identify correctly without a specialist. As AI is becoming more common in healthcare, there is a need for an easy, reliable, and automatic system to detect and classify these aging signs. This project aims to create such a system using deep learning and computer vision.

Expected outcome:

1. **A fully structured facial skin aging dataset** organized into four classes:

- Clear Skin
- Dark Spots
- Puffy Eyes
- Wrinkles

2. **A DataFrame (df)** containing all image filepaths and corresponding labels extracted from folder structure.
3. **Successfully preprocessed images**, resized to 256×256, normalized (0–1 scaling), and stored in an array ready for deep learning models.
4. **Automated data augmentation pipeline** using Keras ImageDataGenerator to improve dataset variability and reduce overfitting.
5. **Label encoding with one-hot vectors**, enabling compatibility with classification models.
6. **Class distribution visualization** for dataset balance analysis and **Augmentation preview visualization** showing how transformations modify sample images.

Tech Stack Used

1. Programming Language

- **Python 3.10.11**

2. Deep Learning Frameworks

- **TensorFlow / Keras**
    - Image loading and augmentation
    - One-hot encoding
    - Data pipeline utilities

3. Image Processing Libraries

- **OpenCV (cv2)**
  - Image reading
  - Resizing
  - Color conversion
- **Pillow (PIL)**
  - Safe image opening
  - Handling corrupted images

5. Visualization Libraries

- **Matplotlib**
  - Augmentation preview
  - Image display
- **Seaborn**
  - Class distribution plots

6. Machine Learning Tools

- **Scikit-Learn**
  - Train/Validation split
  - Data shuffling

7. Development Environment

- **Jupyter Notebook / JupyterLab**

MILESTONE 1: Dataset Preparation and Preprocessing

MODULE 1: Dataset Setup & Image Labeling

Module 1 focuses on preparing the dataset so the model can understand it later. In deep learning, data preparation is one of the most important steps because the model's accuracy completely depends on clean, well-organized data.

1. **Dataset Setup**
   The dataset was structured into four separate class folders — **Wrinkles**, **Dark Spots**, **Puffy Eyes**, and **Clear Skin** — each containing images representing that skin condition.

   ### Code Snippet

   ```
   base_dir = "data"
   classes = ["Wrinkles", "Dark Spots", "Puffy Eyes", "Clear
   Skin"]
   ```

   Folder structure is like:

   ```
   data/
    ├── Wrinkles
    ├── Dark Spots
    ├── Puffy Eyes
    └── Clear Skin
   ```

2. **Building a DataFrame of Filepaths & Labels**

   We create a function to walk through each class folder, pick up image paths, and store them with their correct label.

   This helps convert the raw dataset into a *machine-readable table*.

3. **Creating the Final Dataset Table**

   After collecting the image filepaths and class labels from all four folders, we organized this information into a structured pandas DataFrame. This table acts as the central dataset index, where each row represents one image along with its corresponding label. The DataFrame contains two main columns:

- filepath: the complete directory path to the image file

- label: the class to which the image belongs (Wrinkles, Dark Spots, Puffy Eyes, or Clear Skin)

Code Snippet

```
df = build_df_from_folders(base_dir, classes)

print("Total images:", len(df))
print(df["label"].value_counts())
df.head()
```

Output

```
label         count
Dark Spots    303
Wrinkles      300
Puffy Eyes    300
Clear Skin    300
```

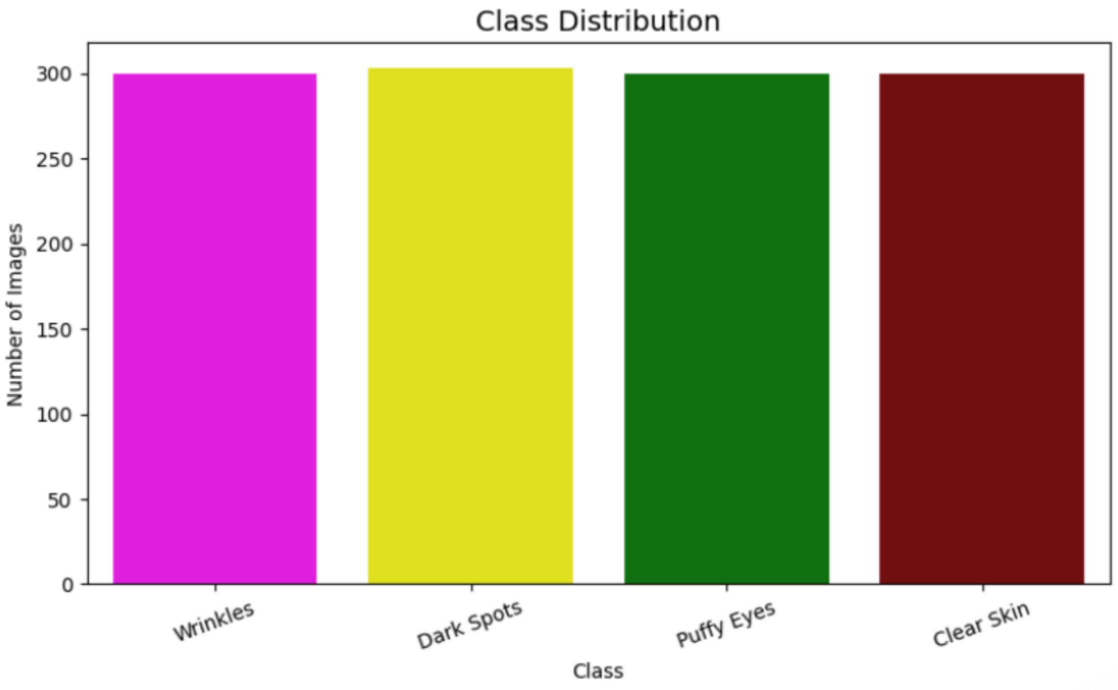
4. Class Distribution

The bar chart shows the number of images available for each skin condition category in the dataset. There are four classes:

- Wrinkles
- Dark Spots
- Puffy Eyes
- Clear Skin

Code Snippet

```
plt.figure(figsize=(8,5))

colors = ["blue", "yellow", "green", "purple"]

sns.countplot(data=df, x="label", hue="label", palette=colors,
legend=False)

plt.title("Class Distribution", fontsize=14)

plt.xlabel("Class")

plt.ylabel("Number of Images")

plt.xticks(rotation=20)

plt.tight_layout()

plt.show()
```



Class Distribution
Number of Images
300
250
200
150
100
50
0
Wrinkles
Dark Spots
Puffy Eyes
Clear Skin
Class