AIDI-1002-01-AI ALGORITHM I

PROJECT-SOW-V1: Human Facial/Expression Recognition

Project Team

Student_ID	Name	Project Role
100820114	Deep Mehta	ALL
100845961	Swati Pal	ALL

Table of Contents

1.1 INTRODUCTION

Facial Expressions plays an important role in interpersonal communication. Facial expression is a non-verbal scientific gesture which gets expressed in our face as per our emotions. Automatic recognition of facial expression plays an important role in artificial intelligence and robotics and thus it is a need of the generation.

1.2 OBJECTIVE OF THE REQUIREMENTS

The objective of this project is to develop Automatic Facial Expression Recognition System which can take human facial images containing some expression as input and recognize and classify it into different expression class such as,

- Happy
- Sad
- Angry
- Confused
- Scared
- Surprised

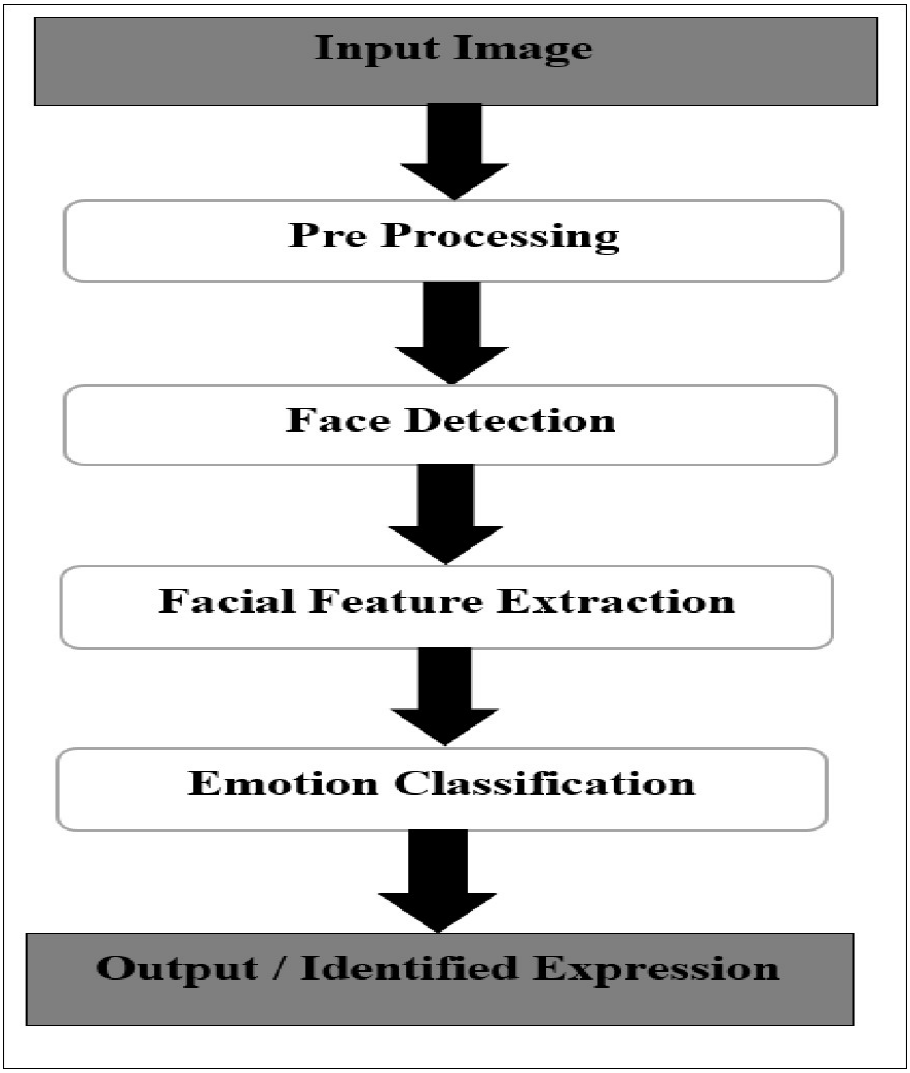
1.3 PROBLEM STATEMENT

Through facial emotion recognition, we are able to measure the effects that content and services have on the audience/users through an easy and low-cost procedure. For example, retailers may use these metrics to evaluate customer interest. Healthcare providers can provide better service by using additional information about patients' emotional state during treatment. Entertainment producers can monitor audience engagement in events to consistently create desired content.

1.4 DETAILED REQUIREMENTS

Facial Expression Recognition System, this can take a person facial images including their expression and emotion as input from user and recognize and do classification into different expression. Numerous Projects are already there related to this subject, but our motivation towards this will not only be to develop an Automatic Facial Expression Recognition System but also to improve accuracy compared to other available systems.

Overall processes of our project:



1.4.1 TASKS, ACTIVTIES, DELIVERABLES AND MILESTONES (WBS)

MILESTONE	ACTIVTIES	TASKS	DELIVERABLE DATE	OWNER	STATUS
REQUIREMENT ANALYSIS	ProjectSetup	GitHub Link	11 Oct,2021	Deep M Swati P	Done
	SOW Creation	Overall layout and prototype building/Data Flow Diagram	25 Oct,2021	Deep M Swati P	Done
	Technical Architecture / Code Review	NA	29 Oct,2021	Deep M Swati P	Not Started
	New Enhancements feasibility analysis	NA	1 Nov,2021	Deep M Swati P	Not Started
DATASET	Analyse Dataset	Importing dataset from Kaggle or from different sources.	3 Nov,2021	Deep M Swati P	Not Started
	Clean Dataset	Training Dataset Testing Dataset	10 Nov 2021	Deep M Swati P	Not Started
DEVELOPMENT	Start	-Develop Data Pre-processing Pipeline -Prototype Algorithm -Develop model*(s)*/architecture	20 Nov,2021	Deep M Swati P	Not Started
	Middle	-Train Model Architecture -Evaluate Model Architecture -Refine Model Architecture	20 Nov,2021	Deep M Swati P	Not Started
	End	Create Application	25 Nov,2021	Deep M Swati P	Not Started

INITIAL UAT	Service	NA	27 Nov,2021	Deep M Swati P	Not Started
	Integration Test	NA	27 Nov,2021	Deep M Swati P	Not Started
BUG RESOLUTION PHASE (OPTIONAL)	UI		02 Dec,2021	Deep M Swati P	Not Started
	Service		02 Dec,2021	Deep M Swati P	Not Started
FINAL UAT	Overall	NA	05 Dec,2021	Deep M Swati P	Not Started
PROJECT DELIVERY			07 Dec,2021	Deep M Swati P	Not Started

1.5 TECHNICAL AND OPERATIONAL ENVIRONMENT

- GIT
- Jupyter Notebook / Google Colab /PyCharm
- Anaconda
- Spyder
- Python
- OpenCV
- Pandas
- Numpy

1.6 ACCEPTANCE CRITERIA

- Comparing different output samples and checking accuracy level

1.7 PROJECT MANAGEMENT CONTROL PROCEFDURES

- Daily 15 minutes' standup call starting from 24 October,2021 until project delivery date
- Weekly update to the professor on the progress until project delivery date
- Fortnightly internal meeting to track the project progress until project delivery date

1.8 CHANGE MANAGEMENT PROCESS

- Any changes to SOW shall be discussed within the team for its feasibility before seeking the professor's approval
- Any agreed changes within the team shall not be made in SOW unless approved by the professor.

1.9 Explanatory Data Analysis

This is pre-processing step to understand the data and find any discrepancies or any other issue in our data. There are various steps involved in the exploratory data analysis. We will get to know the data and explore and extract information to achieve our objective of Human Face Expression Recognition.

Importing the Data:

First step is to import the data and get to know it Importing and Knowing the data. We are importing the data using pandas in Jupyter Notebook.
The dataset we are using for this project is based on human face expressions and has been taken from the Kaggle.

```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sns
        import matplotlib.pyplot as plt
        %matplotlib inline
        sns.set()
```

```
In [2]: df=pd.read_csv("human_face_dataset.csv")
        df.head()
```

Out[2]:

	emotion	pixels	Usage
0	0	70 80 82 72 58 58 60 63 54 58 60 48 89 115 121...	Training
1	0	151 150 147 155 148 133 111 140 170 174 182 15...	Training
2	2	231 212 156 164 174 138 161 173 182 200 106 38...	Training
3	4	24 32 36 30 32 23 19 20 30 41 21 22 32 34 21 1...	Training
4	6	4 0 0 0 0 0 0 0 0 0 0 0 0 3 15 23 28 48 50 58 84...	Training

Data Type:

The variable in our data consists of the following types,

- Emotion: int64
- Pixels: object
- Usages: object

```
In [3]: df.shape
Out[3]: (35887, 3)

In [4]: df.info()
        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 35887 entries, 0 to 35886
        Data columns (total 3 columns):
         #   Column   Non-Null Count  Dtype
        ---  ------   --------------  -----
         0   emotion  35887 non-null  int64
         1   pixels   35887 non-null  object
         2   Usage    35887 non-null  object
        dtypes: int64(1), object(2)
        memory usage: 841.2+ KB
```

```
In [5]: df.describe()
```

Out[5]:

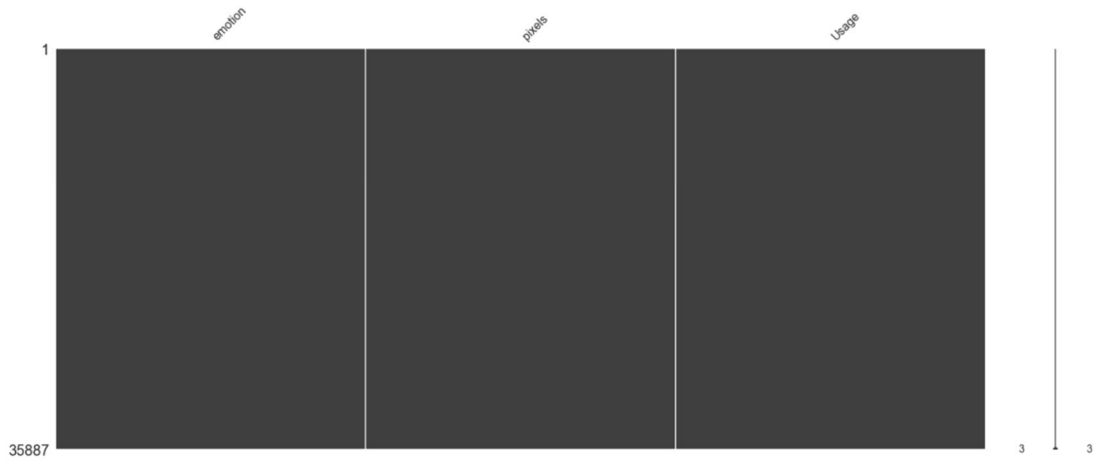
	emotion
count	35887.000000
mean	3.323265
std	1.873819
min	0.000000
25%	2.000000
50%	3.000000
75%	5.000000
max	6.000000

```
In [6]: df.columns
Out[6]: Index(['emotion', 'pixels', 'Usage'], dtype='object')
```

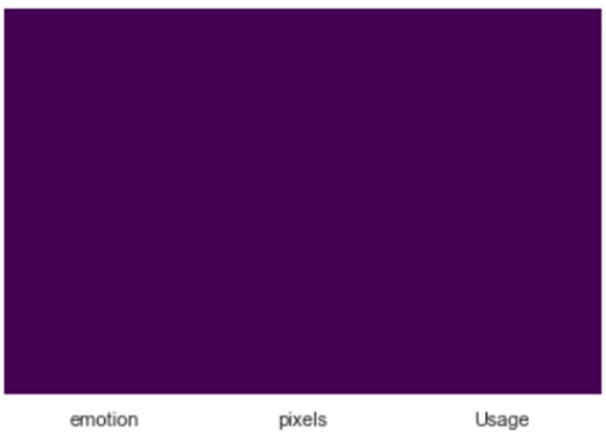
Data Cleaning:

Now we will look if there are any noises in our data

```
In [7]: df.isnull().sum()
Out[7]: emotion    0
        pixels     0
        Usage      0
        dtype: int64
In [9]: import missingno as msno
        msno.matrix(df)
Out[9]: <AxesSubplot:>
```




```
In [14]: sns.heatmap(df.isnull(), cbar=False, yticklabels=False, cmap='viridis')
Out[14]: <AxesSubplot:>
```



Checking for Duplicate Values:

Now we are going check in our data if there are any duplicate rows

```
In [10]: duplicated = df.duplicated().sum()
         if duplicated:
             print("Duplicate rows in Dataset are {}".format(duplicated))
         else:
             print("Dataset contains no duplicate values")
         duplicated = df[df.duplicated(keep=False)]
         duplicated.head()
```

```
Duplicate rows in Dataset are 1234
```

Out[10]:

	emotion	pixels	Usage
30	3	234 233 228 231 234 233 236 230 236 196 112 85...	Training
38	0	255 82 0 3 0 0 0 0 0 0 3 0 16 17 3 60 29 0 1 2...	Training
50	2	99 95 101 114 124 129 128 127 128 124 123 125 ...	Training
62	3	210 161 140 166 159 98 96 141 157 137 110 107 ...	Training
69	5	43 43 43 40 45 63 93 140 144 153 159 167 168 1...	Training

Data Describe:

Now with df.describe(), we are going to check the max min and mean values of our attributes.

```
In [5]: df.describe()
```

Out[5]:

	emotion
count	35887.000000
mean	3.323265
std	1.873819
min	0.000000
25%	2.000000
50%	3.000000
75%	5.000000
max	6.000000

Data Value Count:

```
In [16]: df["Usage"].value_counts()

Out[16]: Training       28709
         PrivateTest     3589
         PublicTest      3589
         Name: Usage, dtype: int64
```

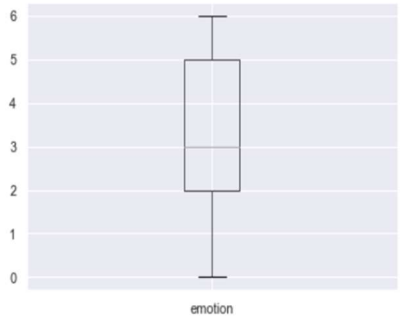
Outliers and Its Removal:

On the examination of data if we find any unusual observations that are far removed from the mass of data. These points are often referred to as outliers.
This shows the boxplot in cholesterol.

In [11]:
```
df['emotion'].unique()
df.emotion.value_counts()
df[df['emotion'] == 4]

df.loc[df['emotion'] == 4, 'emotion'] = np.NaN
df['emotion'].unique()

df.pixels.value_counts()
df[df['pixels'] == 0]

df.loc[df['pixels'] == 0, 'pixels'] = np.NaN
df['pixels'].unique()

df = df.fillna(df.median()) # Replacing with Mean
```

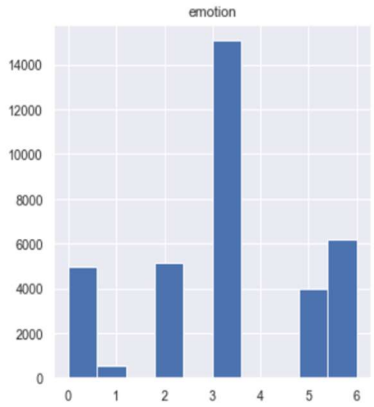
In [12]:
```
df.boxplot()
```

Out[12]: <AxesSubplot:>



Variable Distribution:

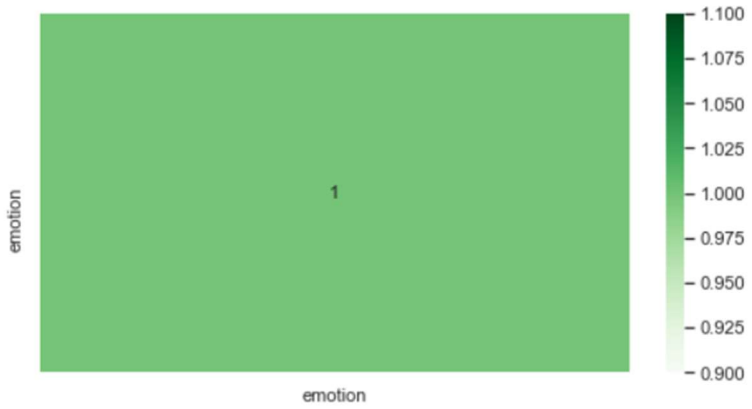
In [18]:
```
#bargrapgh_eda_dist
fig=plt.figure(figsize=(5,5))
ax=fig.gca()
df.hist(ax=ax)
plt.show()
```



Correlation Between Variables:

```
In [15]: plt.figure(figsize=(8,4))
         sns.heatmap(df.corr(),cmap='Greens',annot=True)
```

```
Out[15]: <AxesSubplot:>
```



Data Splitting:

Now we are going to split our data in two training data set and test dataset where we will use our algorithm to check our prediction. As it is important to not to run the test on the training data to get the accurate result. We used the “train_test_split” function in Scikit-learn to split our data. We have used Standard Scalar to fit the data as we want to bring all the measurements into a single scale to help the performance. Standard Scaler subtracts the mean value and then converting it to unit vector. We get unit variable by dividing the variable values with standard deviation.

```
In [21]: from sklearn.model_selection import train_test_split

         #### Training and Testing data:

         df_train,df_test = train_test_split(df,train_size=0.7,random_state=50)

         y_train = df_train.pop('emotion')
         X_train = df_train

         y_test = df_test.pop('emotion')
         X_test = df_test

         print(X_train)
         print(y_train)
```

```
                                                  pixels        Usage
27683  61 70 83 93 81 84 107 95 108 111 110 120 110 1...     Training
6731   87 85 95 107 97 106 86 90 111 102 61 44 39 42 ...     Training
24202  251 253 255 255 255 255 254 255 254 208 158 14...     Training
35482  221 218 217 215 212 206 188 195 197 173 161 16...  PrivateTest
25696  208 209 209 212 215 218 215 210 206 201 203 13...     Training
...                                                  ...          ...
8559   54 153 246 226 238 228 232 234 222 219 233 234...     Training
34887  180 190 189 196 197 197 197 196 196 194 174 15...  PrivateTest
32022  120 121 119 117 119 119 119 121 120 131 184 21...   PublicTest
22637  216 224 134 82 100 197 175 81 70 78 74 76 82 8...     Training
14000  254 254 253 253 250 255 222 102 77 69 61 64 61...     Training

[25120 rows x 2 columns]
27683    6.0
6731     0.0
24202    2.0
35482    2.0
25696    2.0
        ...
8559     3.0
34887    0.0
32022    3.0
22637    2.0
14000    6.0
Name: emotion, Length: 25120, dtype: float64
```

Authorization:

This scope has been authorized and approved.

Date and Signature