|  |  |
| --- | --- |
| Name | ID |
| Moman Mahamed | 20210976 |
| Mariam Ahmed abo ELfadl | 20210905 |
| Warda Khaled | 20211038 |
| Mira Hesham Helmy | 20210980 |
| Abdallah Samy | 20210547 |
| Abdelrahaman Mahmoud | 20210530 |

Image Dataset:

Dataset Name: ***Facial Expression Recognition:***

* ***Dataset:***[***Facial Expression Recognition Challenge (FER2013)***](https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data)

**Image Dataset:**

[**https://www.kaggle.com/datasets/msambare/fer2013/code**](https://www.kaggle.com/datasets/msambare/fer2013/code)

About Dataset:

**The data consists of 48x48 pixel grayscale images of faces. The faces have been automatically registered so that the face is more or less centred and occupies about the same amount of space in each image.**

**The task is to categorize each face based on the emotion shown in the facial expression into one of 5 categories**

**(0=Angry, , 1=Happy, 2=Sad, 3=Surprise, 4=Neutral).**

***Classes and their labels:***

* 5 classes [happy ,sad, angry ,neutral ,surprise]

***Total no of samples in dataset and size of each***

* 30,219 Sample,117 mb
* Number of samples in the training set: 24,176
* **Number of samples in the testing set: 6,043**

**Implementation details :**

**We used two algorithms K-Means and Logistic Regression**

**Feature Extraction Phase:**

**Histogram of Oriented Gradients (HOG) in Logistic Regression**

**And ORB In KMeans**

**The feature extraction phase in this project employs the Histogram of Oriented Gradients**

**(HOG) technique to extract discriminative features from images.**

Class names are encoded into numeric labels using label\_encoder.

Data is divided into training (X\_train, y\_train) and testing (X\_test, y\_test) sets.

into 20% test and 80% train by train\_test\_split algoritm.

train\_data stores training features and labels, while test\_data holds testing features and

labels.

• Training Data:

• train\_data combines features (X\_train) and their respective labels (y\_train).

• X\_train holds features used for model training.

• y\_train contains corresponding labels for the features in X\_train.

• Testing Data:

• test\_data merges features (X\_test) and their associated labels (y\_test).

• X\_test includes features for testing the trained model.

• y\_test stores labels linked to the features in X\_test

**KMEANS**

The code loads, preprocesses, and visualizes grayscale images. It extracts ORB descriptors, applies k-means clustering, and creates a bag-of-words representation. The descriptors are clustered using PCA for 2D visualization. The implementation utilizes OpenCV, NumPy, Matplotlib, Seaborn, and scikit-learn libraries for image processing, feature extraction, and clustering in a comprehensive image analysis pipeline and we use ORB as Feature Extraction , which stands for Oriented FAST and Rotated BRIEF, is a feature detection and description algorithm in machine learning and computer vision. It was introduced as an efficient alternative to traditional methods like SIFT and SURF. ORB combines the FAST keypoint detector for identifying interest points with the BRIEF binary descriptor for describing the local features around those points. It is particularly suitable for real-time applications due to its computational efficiency and ability to handle rotation variations.

**Logistic Regression**

Goal:

The goal of this code is to perform a classification task using Logistic Regression on

Histogram of Oriented Gradients (HOG) features. It aims to evaluate model performance,

including accuracy metrics, confusion matrix visualization, and decision boundary plotting.

Code Explanation:

Data Preparation:

Scales the HOG features using StandardScaler and applies PCA for visualization.

Defines class labels and initializes Logistic Regression for a multiclass classification problem.

Model Training and Evaluation:

Fits the Logistic Regression model on the scaled HOG features and evaluates its accuracy on

the entire dataset.

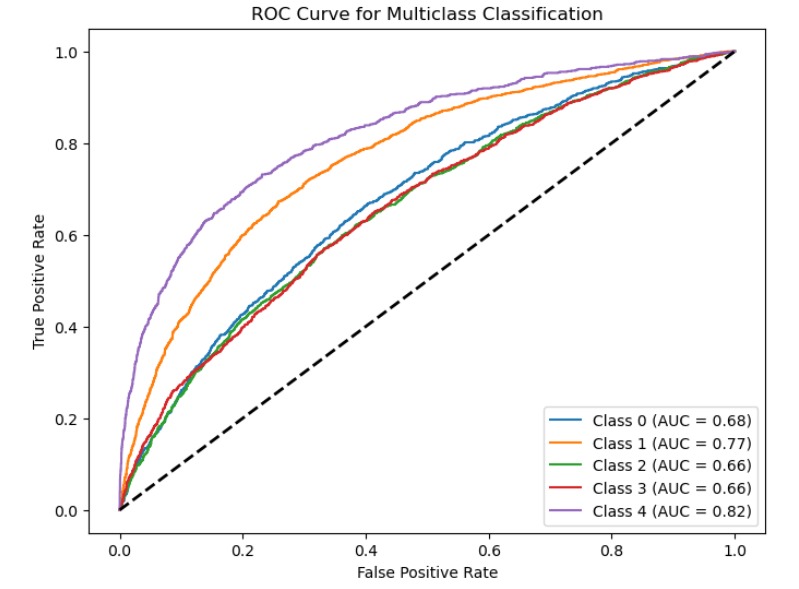
Confusion Matrix Visualization:

Creates a confusion matrix for the entire dataset and visualizes it using a heatmap.

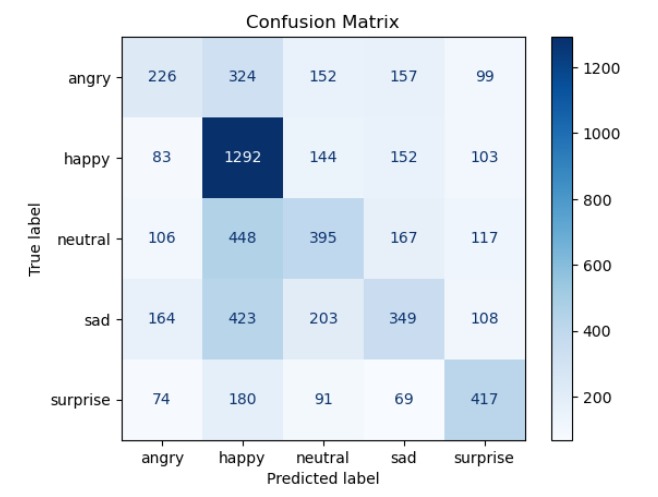
Batch Size:Default Size

No of epochs:100

**ROC Curve**

****

**Confusion Matrix**

****

**G**) **Accuracy:** 0.44332285288763856

Numerical Dataset:

**1)Numerical Dataset : Linear & Knn model**

**A)General Information on dataset**

* Dataset Name: California Housing data [California Housing Data (kaggle.com)](https://www.kaggle.com/datasets/dhirajnirne/california-housing-data)

**About Dataset**

**Context**

This is the dataset used in the second chapter of Aurélien Géron's recent book 'Hands-On Machine learning with Scikit-Learn and TensorFlow'. It serves as an excellent introduction to implementing machine learning algorithms because it requires rudimentary data cleaning, has an easily understandable list of variables and sits at an optimal size between being to toyish and too cumbersome.

The data contains information from the 1990 California census. So although it may not help you with predicting current housing prices like the Zillow Zestimate dataset, it does provide an accessible introductory dataset for teaching people about the basics of machine learning.

**Content**

The data pertains to the houses found in a given California district and some summary stats about them based on the 1990 census data. Be warned the data aren't cleaned so there are some preprocessing steps required! The columns are as follows, their names are pretty self explanitory: longitude,latitude,housing\_median\_age,total\_rooms,,total\_bedrooms,population,households,median\_income,median\_house\_value,ocean\_proximity

* Total number of samples in the dataset : 20640

After preprocessing (cleaning and removing outliers)

* Total number of samples in the dataset After preprocessing: 18582
* Number of samples in the training set: 14865
* Number of samples in the testing set: 3717

**B)Implementation Details**

*At feature extraction phase:*

* Number of features before Hot encoding: 9
* Number of features after Hot encoding: 13
* 5 Features got extracted after encoding ['ocean\_proximity\_<1H OCEAN', 'ocean\_proximity\_INLAND', 'ocean\_proximity\_NEAR BAY', 'ocean\_proximity\_NEAR OCEAN', ‘ocean\_proximity\_ISLAND']
* Feature names after encoding: ['longitude', 'latitude', 'housing\_median\_age', 'total\_rooms', 'total\_bedrooms', 'population', 'households', 'median\_income', 'ocean\_proximity\_<1H OCEAN', 'ocean\_proximity\_INLAND', 'ocean\_proximity\_ISLAND', 'ocean\_proximity\_NEAR BAY', 'ocean\_proximity\_NEAR OCEAN']
* Dimension of Resulting Features (Number of Samples, Number of Features): (18582, 14)
* We use StandardScaler, StandardScaler is a preprocessing technique used to standardize or scale the features of a dataset. It transforms the data by subtracting the mean and dividing by the standard deviation of each feature, ensuring that the features have a mean of 0 and a standard deviation of 1. This process helps in normalizing the data, making it suitable for algorithms that are sensitive to the scale of input features, such as support vector machines or k-nearest neighbors. StandardScaler is often applied to numerical features to improve the performance and convergence of machine learning models.
* In the KNN model, the key hyperparameter is the number of neighbors (n\_neighbors), which is set to 5

**C) *Results details:***

**Goal:**

To predict House prices

Linear Regression

Training Set Metrics:

* **R-squared Score: 0.663811455138456**
* **Mean Absolute Error (MAE): 48991.273756497336**
* **Mean Squared Error (MSE): 4483054989.618424**
* **Root Mean Squared Error (RMSE): 66955.61955219609**

Test Set Metrics:

* **Mean Absolute Error (MAE): 49037.18881782961**
* **Mean Squared Error (MSE): 4571763432.421956**
* **Root Mean Squared Error (RMSE): 67614.81666337606**

**R-squared Score: 0.6625598943848602**

**A purple dotted line graph

Description automatically generated with medium confidence**

Knn (K\_Neighbours)

**k-NN Training Set Metrics:**

* R-squared Score: 0.818052346435404

k-NN Test Set Metrics:

* **Mean Absolute Error (MAE): 40624.01549636804**
* **Mean Squared Error (MSE): 3701536092.7199783**
* **Root Mean Squared Error (RMSE): 60840.25059711686**
* **R-squared Score: 0.726791040584534**

A purple dotted line with white text

Description automatically generated with medium confidence

A chart with orange and blue dots

Description automatically generated