Performance Analysis of a set of five ML and DL Algorithms on FER2013 Dataset

Subhiksha S.†, Supraja S.†, Umarani K.†, Vaishnavi Anand†
Department of Information Technology
Batch of 2020
SSN College of Engineering
Kalavakkam, Chengalpattu, TN, India
{subhiksha2010676, supraja2010341, umarani2010455, vaishnavi2010276}@ssn.edu.in

ABSTRACT

This paper depicts the performance of a set of 5 machine learning (ML) and deep learning (DL) models—CNN, GNB, kNN, MLP, mini-batch k-Means with PCA, run on FER2013 benchmark dataset and includes the inferences drawn from it for Facial Emotion classification. The FER2013 dataset is a labeled dataset for headshots depicting emotions in seven classes (happy, sad, angry, afraid, surprise, disgust, and neutral). The ML and DL models chosen above are used to determine the best possible model for FER. Using the test dataset of FER2013, we were able to obtain an accuracy score of 61.27% accuracy using CNN, 25% using MLP, 30% from mini-batch k-means, 23% from GNB and 51% using kNN classification algorithms. Performance metrics, hyperparameters and other heuristic methods are tuned to try improving accuracy, and conclusions are drawn from the performance of the models.

Index Terms - Exploratory Data Analysis (EDA), Convolutional Neural Networks (CNN), Naive Bayes (NB), k-Nearest Neighbours (kNN), Multi-layer Perceptron (MLP), k-Means with Principal Component Analysis (PCA), Facial Emotion Recognition (FER), FER2013, Machine Learning (ML), Deep Learning (DL), minibatch K-means

I. INTRODUCTION

Algorithms that learn from structured data to predict outputs and discover patterns in that data are called Machine Learning Algorithms whereas algorithms based on highly complex neural networks that mimic the way a human brain works to detect patterns in large unstructured data sets are called Deep Learning Algorithms. Each algorithm has a specialty of its own such as CNN classifies image dataset and RNN classifies text-based dataset better than other datasets if given to these algorithms.

Facial Emotion Recognition (FER) analysis comprises three steps: a) face detection, b) facial expression detection, c) expression classification to an emotional state. Emotion detection is based on the analysis of facial landmark positions (e.g. end of nose, eyebrows). FER is commonly done on static images or videos for sentiment analysis and finds scope in various fields, for instance, customer satisfaction analysis, crime detection, healthcare and education.

The Emotion Detection dataset contains 35,685 examples of 48x48 pixel gray scale images of faces divided into train and test dataset. Images are categorized based on the emotion shown in the facial expressions (happiness, neutral, sadness, anger, surprise, disgust, fear). The faces have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image. The FER2013 dataset is a .csv file of the Emotion Detection dataset.

Naïve Bayes algorithm is a supervised learning algorithm, which is based on **Bayes theorem** and used for solving classification problems. Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building fast machine learning models that can make quick predictions. It is a probabilistic classifier and belongs to Supervised Learning. It is based on probability models that incorporate strong independence assumptions. The independence assumptions of the Naive Bayes models often do not impact reality. Therefore, they are considered naive. The two variants used are Bernoulli and Gaussian Naïve Bayes Classifiers.

Multilayer Perceptron (MLP) is a type of feedforward neural network that consists of multiple layers of nodes, including an input layer, one or more hidden layers, and an output layer. These layers are fully connected, and the algorithm is used for classification.

A Convolutional Neural Network is a type of deep learning algorithm commonly used for image classification and video recognition tasks. CNNs are designed to automatically learn relevant features from raw pixel data, through a series of convolutional and pooling layers. These layers learn to recognize low-level features like edges and corners and then combine these features into higher-level representations of objects and scenes. The output of the network is fed into fully connected layers to perform classification or regression tasks. CNNs have shown impressive performance on wide range of image and video recognition tasks, and are widely used in fields such as computer visions, robotics, and automatic vehicles.

† These authors share equal authorship of this work and respective contributions have been listed below.

K-nearest neighbours (kNN) is a naïve, non-parametric supervised algorithm that works well with binary as well as multi-class classification problems. kNN keeps all the training data to make future predictions by computing the similarity between an input sample and each training instance. Its working can be summarized as I) computes the distance between the new data point with every training example. II) For computing distance, measures such as Euclidean distance, Hamming distance or Manhattan distance will be used. III) Model picks K entries in the database which are closest to the new data point. IV) Then, the majority vote i.e. the most common class/label among those K entries will be the class of the new data point.

k-means clustering is popular, hard clustering algorithm (unsupervised) that divides n observations into k predefined non-overlapping clusters / sub-groups where each data point belongs to only one group. K-means generally works well with spatial data like images and is a centroid-based algorithm where we assign a centroid to a cluster and the whole algorithm tries to minimize the sum of distances between the centroid of that cluster and the data points inside that cluster. Here, we use a variation of classic k-means called mini-batch k-means. The working of mini-batch k-means can be summarised as using small random batches of data of a fixed size, so the data can be stored in memory. During each iteration, a new random sample from the dataset is obtained and used to update the clusters and this is repeated until convergence. It is popularly used with high-dimensional datasets. We observe the number of clusters that is optimal for FER dataset without considering original labels.

II. FER2013 and Emotion Detection Datasets

In fer2013 dataset and Emotion Detection dataset, each face is categorized based on the emotion shown in the facial expression into one of seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral). The training set consists of 28,709 examples and the public test set consists of 3,589 examples.

A. Content

1) Emotion: Each image is classified into categories of emotions (0=Angry, 1=Disgusted, 2=Fearful, 3=Happy, 4=Sad, 5=Surprised, 6=Neutral).

2) Pixels: An object consisting of each pixel value of an image in a string which has 34034 unique values.

3) Usage: It is divided into 3 sections which has 80% of data in the training section and 10% in the Public Test section and 10% in the Private Test section.

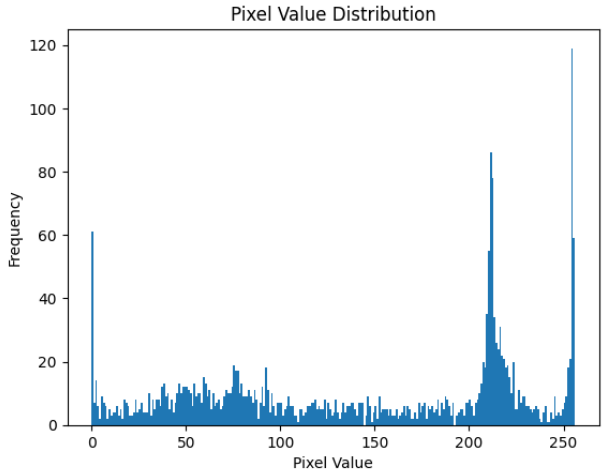
Greyscale images are divided into test and training folders and inside both the folders, images are grouped under the folders of angry, disgusted, fearful, happy, sad, surprised and neutral.

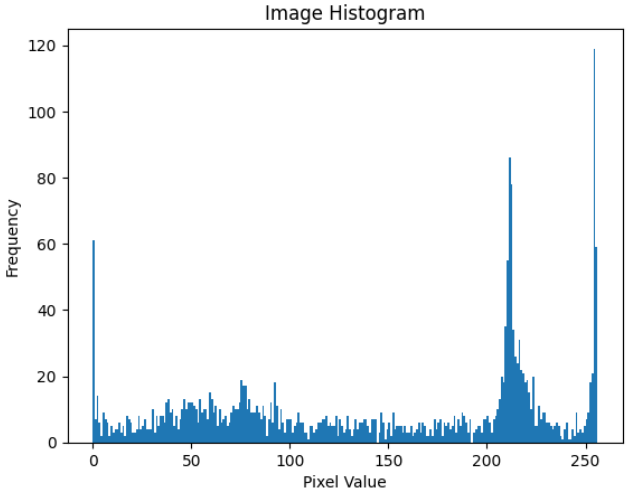
III. Exploratory Data Analysis

Exploratory Data Analysis (EDA) is an approach that is used to analyze data and discover trends, patterns, or check assumptions in data with the help of statistical summaries and graphical representations.

A. EDA on images

1) *Reading an image and displaying:* We extract the image using the path of the image. We make use of functions from the io library to perform this.
2) *Computations:* We compute image properties and statistics using the functions in the numpy library.
3) *Graphical Representation:* Pixel Value Distribution and Image histogram were represented using histograms.

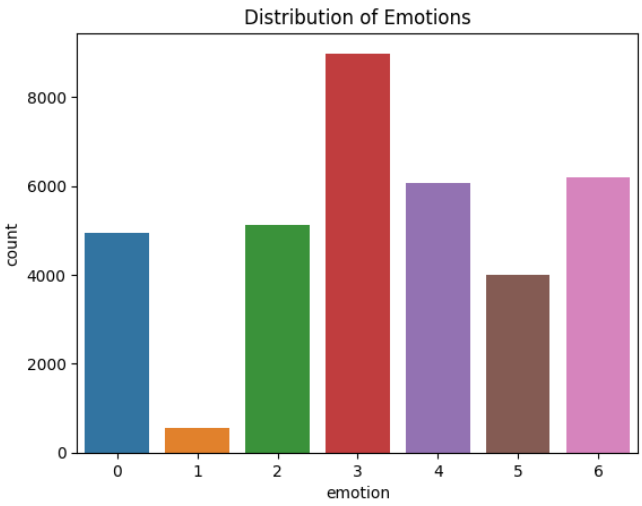


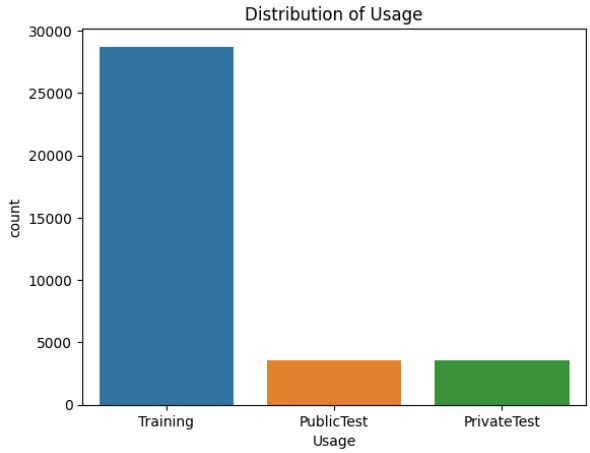


4) *Mean Image:* Computing the mean image of a particular class and displaying it

B. EDA on csv file

1) *Data features:* We display few records of data to get the features involved in the data and how they are represented. We display the number of records and attributes.
2) *Graphical Representation:* We make use of bar graph to display the Distribution of Emotions and Distribution of Usage.





C. Data Preprocessing

We need numerical type of data to process models. Since the pixels are in object type we convert it into float array using numpy library. We split the data into X_train, y_train, X_test and y_test which are necessary for models to train and test. We check if they are done correctly by printing some records. In FER2013, usage labels for records are already provided i.e. Training, PrivateTest and PublicTest. These labels are used for splitting the entire dataset and reducing dimensionality.
We further split the training data ("Training" usage in FER2013) as training and validation sets used in training and validating some models.

IV. ML AND DL MODELS FOR FER2013 DATASET

CNN:

Convolutional Neural Networks (CNNs) are a type of deep learning algorithm that are commonly used in computer vision tasks, including emotion detection from images or videos. Emotion detection is the process of automatically recognizing and classifying the emotional state of a person from their facial expressions.

A typical CNN architecture for facial emotion recognition would consist of several layers, including convolutional layers, pooling layers, and fully connected layers. The convolutional layers are responsible for extracting features from the input image, while the pooling layers reduce the spatial dimensions of the feature maps. The fully connected layers then use these features to classify the image into one of several emotion categories or classes . In this FER 2013 dataset we have 7 classes that are angry, disgust, fear, happy, neutral, sad, surprise..

To train a CNN for Facial Emotion Recognition , a large dataset of labeled images of FER2013 dataset is used. This dataset contain a variety of different facial expressions, as well as annotations indicating the corresponding emotion for each image.

During training, the CNN learns to recognize patterns in the input images from the training set of images that are associated with specific emotions. This is accomplished through a process of forward propagation, where the input image is passed through the layers of the network.

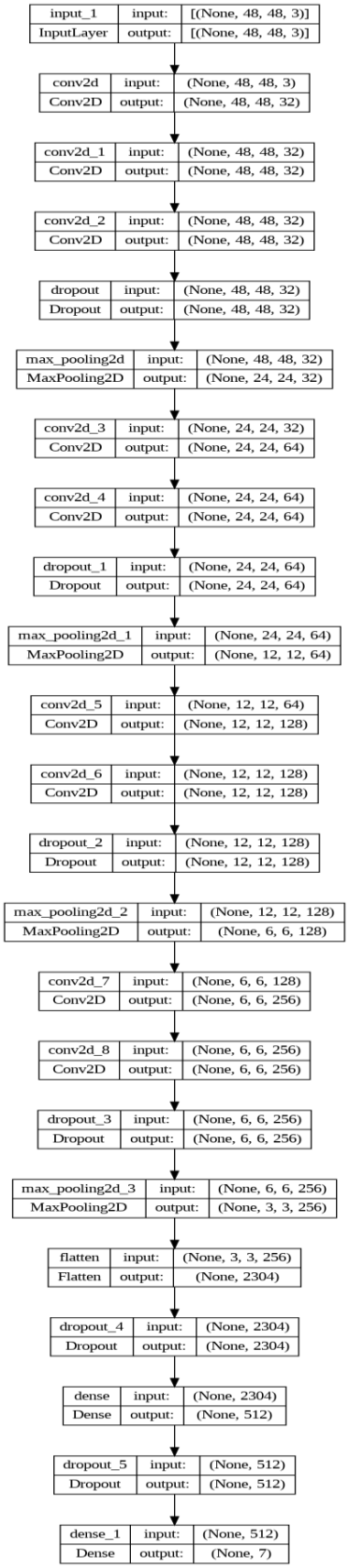
The convolutional layer performs a mathematical operation called convolution on the input image, which applies a filter to each small region of the image to extract relevant features such as edges and textures.

The output of the convolutional layer is then passed through a pooling layer, which reduces the dimensionality of the feature maps by selecting the most relevant features. The most common type of pooling is max pooling, which selects the maximum value from each small region of the feature map. Max pooling is effective at preserving the most important features while reducing the computational complexity of the network. Then the output of these two layers given to the fully connected network.

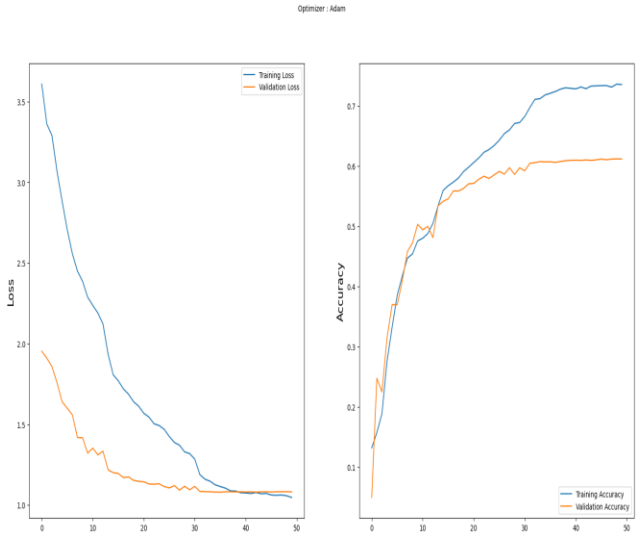
The Adam optimizer updates the weights of the network using the gradients of the loss function with respect to the weights. It does this by computing a moving average of the gradients and the squared gradients, and using these to adjust the learning rate for each weight. This adaptive learning rate helps to ensure that the weights converge to the optimal values quickly and accurately.

The resulting output is compared to the true emotion label using a loss function. The weights of the network are then adjusted using backpropagation in order to minimize the loss function and improve the accuracy of the network.

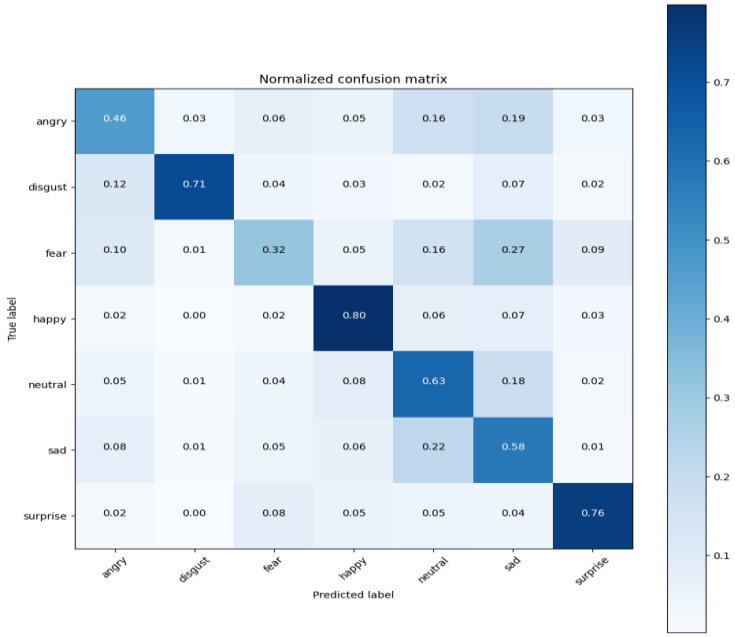
RESULT OF CONVOLUTION AND POOLING LAYER:



PLOTTING ACCURACY AND LOSS :



CONFUSION MATRIX :



Accuracy provided by the CNN classifier: 0.61193

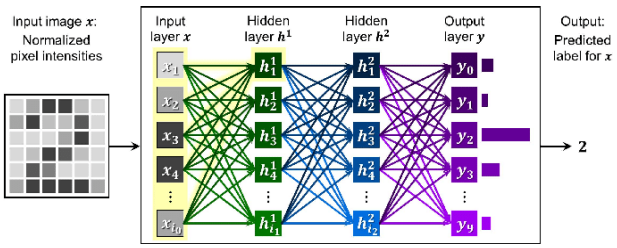
From the accuracy we see that CNN model accuracy can be improved for an image dataset as accuracy is still low. This may be because the human face can be a little bit confusing when expressing emotions like surprise, or complex neutral expressions which can lead to more overlap between the images' classes.

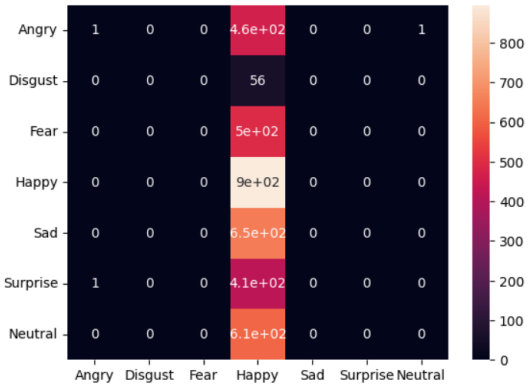
From the above visualization of confusion matrix also we can see that for surprise, happy, disgust classes, the true negative rate is high. So we show the model is well trained for happy, disgust, surprise images.

MLP:

MLP was chosen for the FER2013 dataset due to its ability to effectively handle multi-class classification problems. With a relatively simple architecture, MLPs can be trained efficiently on large datasets, making them a suitable choice for the image recognition task presented by the FER2013 dataset. Additionally, MLPs can provide a good baseline for comparison with more complex models such as CNNs, allowing for a better understanding of the dataset and problem at hand.

MLP is used to classify emotions based on the features extracted from the images. The input layer takes the flattened pixel values of the image as input, and the output layer gives the probability distribution of each class. MLP was implemented in a few ways on the dataset to look for changes in accuracy. The MLP classifier is an inbuilt class, was implemented on the dataset with only 1 hidden layer with 128 nodes. Secondly, an MLP model with 7 different layers, each for an emotion was developed. This model was tried to be refined with image augmentation to try to improve accuracy. All the models, however, gave a standard accuracy of 25%. This constant accuracy, despite possible accuracy boosting techniques indicates that the accuracy cannot be improved further for MLP.





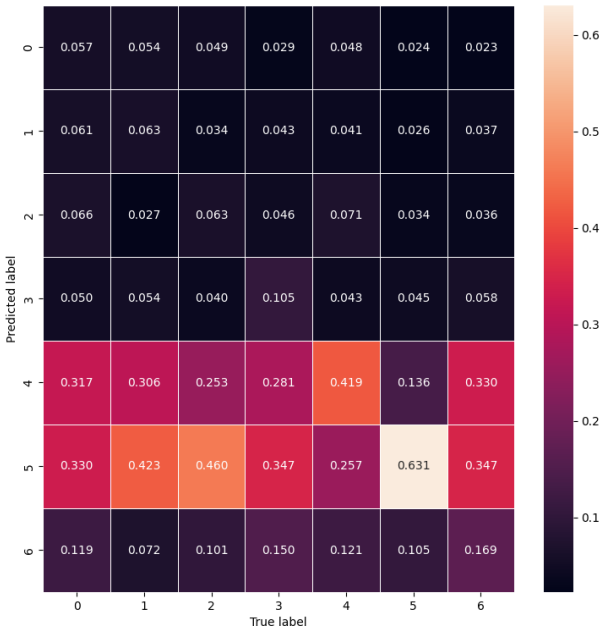
Naïve Bayes Classifier:

Naive Bayes classifier is used for well separated categories, since fer2013 dataset has categories well separated it was possible to use this classifier. Naive Bayes classification is extremely fast for training and prediction especially using logistic regression.
It provides straightforward probabilistic prediction and has an extremely low computation cost. It can efficiently work on a large dataset and can be used with multiple class prediction problems.
Thus, it was implemented in fer2013 benchmark dataset which is large and a multi class prediction problem. Two most important variants of this classifier were run on the dataset which are the following:

The **Bernoulli Naive Bayes** (BNB) is a part of the family of Naive Bayes. It only takes binary values. Multiple features may exist, but each is assumed to be a binary-valued (Bernoulli, boolean) variable. Therefore, this class requires samples to be represented as binary-valued feature vectors.
Bernoulli Naive Bayes Classifier is based on the following rule:

$$P(x_i \mid y) = P(i \mid y)x_i + (1 - P(i \mid y))(1 - x_i)$$

According to the decision rule formula, x needs to be binary. Think about the formula in the case where xi=1 and the case where xi=0. So i is the event where xi=1 or the event where xi=0.
However when implemented on the fer2013 dataset it gives a low accuracy and has been able to classify only 2 classes properly as shown below in the form of confusion matrix that will visually help us see the number of correct and incorrect classified classes,



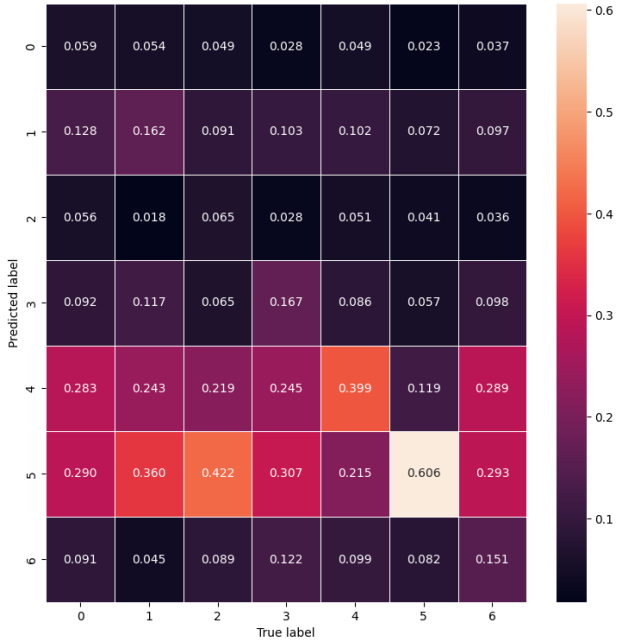
Accuracy provided by the BNB classifier:
0.21830593480078017

The **Gaussian Naive Bayes** (GNB) is a variant of Naive Bayes that follows Gaussian normal distribution and supports continuous data. To build a simple model using Gaussian Naive Bayes, we assume the data is characterized by a Gaussian distribution with no covariance (independent dimensions) between the parameters. This model may fit by applying the Bayes theorem to calculate the mean and standard deviation of the points within each label.
If we assume that X's follow a Gaussian or normal distribution, we must substitute the probability density of the normal distribution and name it Gaussian Naïve Bayes. To compute this formula, you need the mean and variance of X.

$$P(X|Y=c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{\frac{-(x-\mu_c)^2}{2\sigma_c^2}}$$

In the above formulae, sigma and mu is the variance and mean of the continuous variable X computed for a given class c of Y.

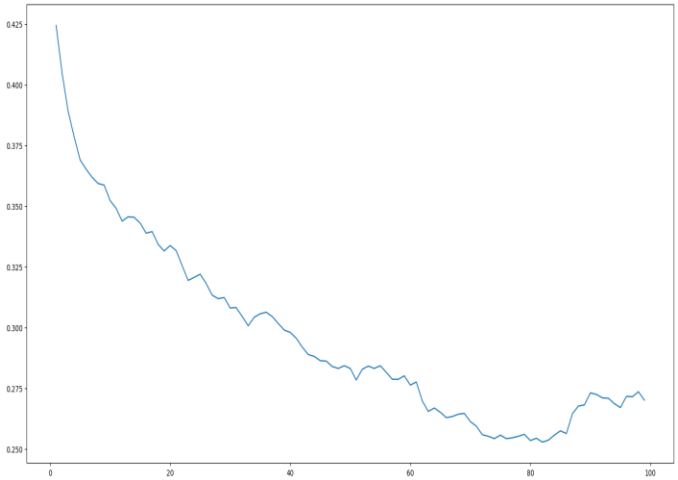
However when implemented on the fer2013 dataset it gives a low accuracy and has been able to classify only 2 classes properly as shown below in the form of confusion matrix that will visually help us see the number of correct and incorrect classified classes,



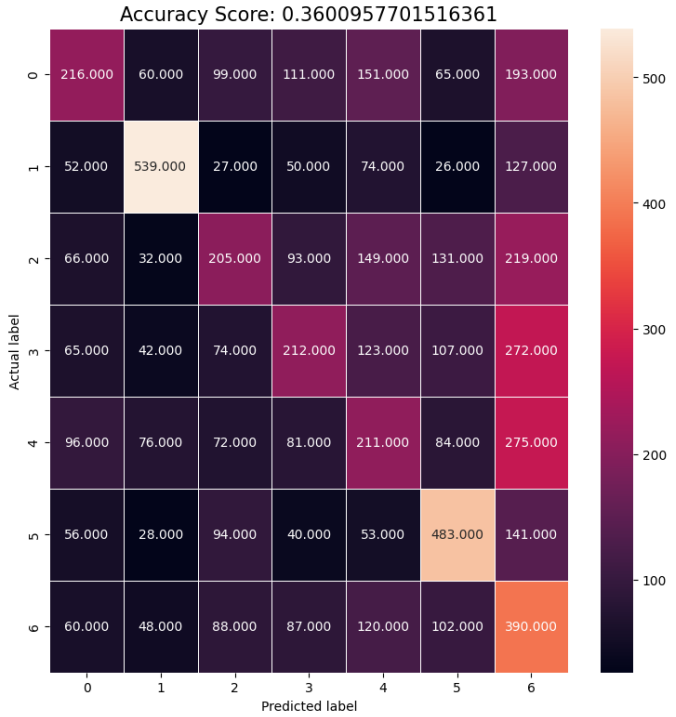
Accuracy provided by the GNB classifier:
0.22638617999442742

The conclusion is that Naïve Bayes Classifier is not efficient to classify the fer2013 dataset.

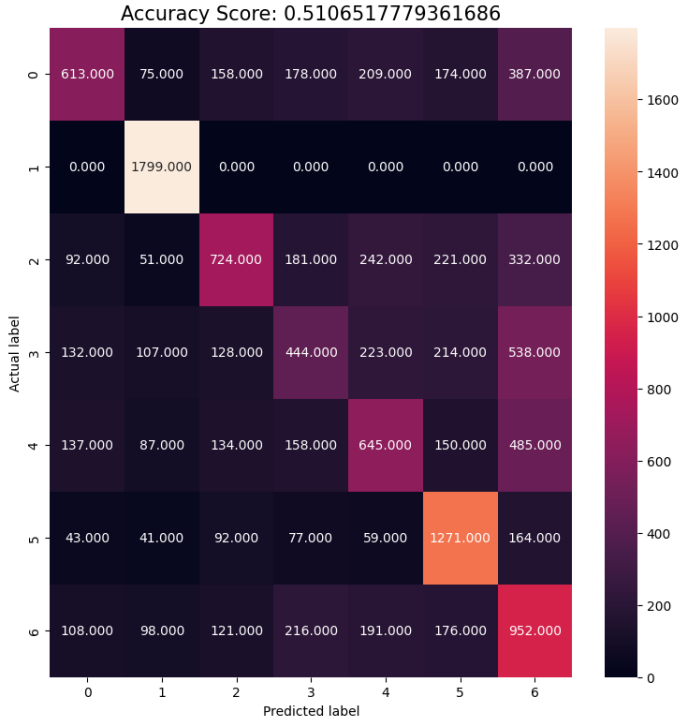
kNN classifier:
The K-nearest neighbour algorithm works on the principle of majority voting where a data point's k-nearest neighbours are found and the most common class label among those is chosen as the label for considered data point. The FER2013 dataset contains labeled images that are converted into pixel matrices. From EDA we see that the instance count plot of the seven classes of images in the original dataset shows that there is an imbalance in data, hence this may influence the decision making of kNN classification. So we use an oversampling technique to balance out minority classes and then split the training dataset into sub-training, test and validation sets. After training and validating the model, we run the same on test FER dataset.
Starting with a k value of 1, we observe that the model returns an accuracy of 42.4%. The hyperparameter we tune is the value of k i.e. the number of neighbours considered. Taking k between 1 to 100, we plot the respective accuracy values to get a graph.



The conclusion is that k=1 gives highest accuracy possible with k-means, hence the nearest neighbour is the most influential in our image classification according to kNN. The algorithm being naïve, non-parametric and lazy, we can say it outperforms expectations by giving an accuracy of 42%.
As we see majority voting in play here, another strategy for multi-class classification, one vs rest is experimented with kNN and we observe if it classifies better. OneVsRest (OvR) classifiers are used with the idea of extending inherently binary classification algorithms like logistic regression to classify in a multi-class scenario. To do this, every class is taken with respect to all the rest thus forming 2 superset classes (here, we consider class [0] vs [1,2,3,4,5,6] and so on). The new strategy gives an output accuracy of 36% and the following confusion matrix on predicted labels.

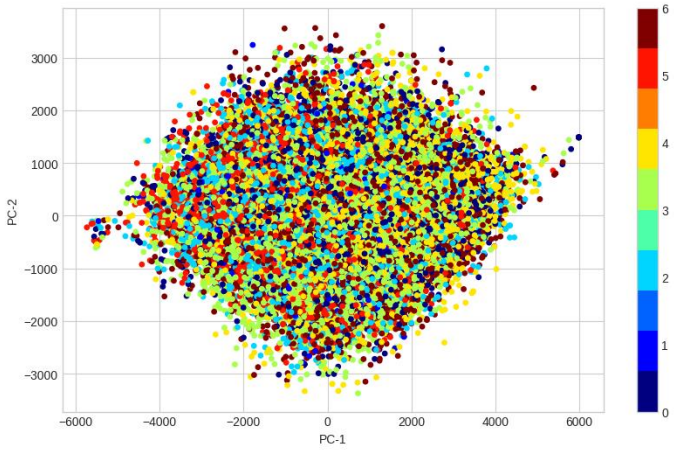


The confusion matrix shows that class 5 has had a high number of true positive cases while the other classes are much lower in accuracy. Similar to this, we visualize the confusion matrix for the testing dataset got by splitting training data and get an accuracy of 51% which is much higher. The algorithm having memorized training data shows more success for a portion of the training dataset itself.



PCA and Minibatch k-means clustering and prediction:

The PCA dimensionality reduction technique for FER dataset is used to reduce the dataset column dimensions from 3 to 2 by considering the 2 most dominant features or correlated features. PCA works by standardizing the range of variables, considering covariance matrix and computing the eigenvalues and eigenvectors for each column (or feature). The FER dataset consists of grayscale images with intensity values taken as pixels (between 0 and 255) and also the class labels along with usage information (whether for training, private test or public test purposes). These are the variables considered. The output of PCA with 2 PCs is shown by a scatterplot.



A scatter plot from PCA must show the clusters by their similarity but here, by eyeballing we can say no clear clusters exist in our dataset or there is necessity for complex clustering/classification methods using DL algorithms.

k-means clustering algorithm is used with FER to label the dataset using the number of clusters as seven (as given by original dataset) and applying k-means. In minibatch k-means, the centroids are assumed by the algorithm for a small fraction of the entire dataset and updated in each iteration till convergence.

Minibatch k-means was chosen as the FER dataset has high-dimensionality with number of training instances exceeding 20,000 and number of pixels considered for testing is 37,878. Reducing dimensionality via PCA with principal components as 2 does not yield suitable results for k-means. To still analyse the performance, we go for minibatch k-means on the test set formed by splitting training dataset.

The algorithm uses the sub-training dataset split from FER training dataset and predicts labels for seven clusters with an accuracy of 19.5%. To improve the stark low accuracy, we go for optimization by tuning the parameter k i.e. the number of clusters taken.

```
Number of clusters is 4
Inertia : 3877825.5
Homogeneity : 0.013550036901865129
Accuracy score : 0.1926976081102487

Number of clusters is 8
Inertia : 3497851.25
Homogeneity : 0.017261314230683125
Accuracy score : 0.19998415967052113

Number of clusters is 10
Inertia : 3383480.5
Homogeneity : 0.018805556423197114
Accuracy score : 0.20038016790749247

Number of clusters is 13
Inertia : 3282949.0
Homogeneity : 0.020441029810177355
Accuracy score : 0.20420824753154865

Number of clusters is 20
Inertia : 3111492.5
Homogeneity : 0.023589479873704355
Accuracy score : 0.20901314747346744

Number of clusters is 36
Inertia : 2940026.0
Homogeneity : 0.028587897853857003
Accuracy score : 0.21796293362901947

Number of clusters is 48
Inertia : 2864481.5
Homogeneity : 0.034793271552809174
Accuracy score : 0.22514388299276625

Number of clusters is 64
Inertia : 2770362.5
Homogeneity : 0.03929392416349252
Accuracy score : 0.23187602302127885

Number of clusters is 144
Inertia : 2578718.75
Homogeneity : 0.07230330319343464
Accuracy score : 0.25993980674798034

Number of clusters is 256
Inertia : 2432472.0
Homogeneity : 0.10995390034127407
Accuracy score : 0.29064364538782406
```

From the above output we observe that the number of clusters with highest accuracy was 256. This may be because FER dataset contains headshots of people in different poses, orientation of the eyes or head may differ, intensity variations, face structures and so on. The accuracy of FER testing dataset (PublicTest usage) for k=256 is 30%. We successfully improved the accuracy score and also used other performance metrics like inertia and homogeneity score to assess and inferred the following trends for FER dataset:

i) the accuracy increases as k value increases

ii) the inertia value decreases as number of clusters reaches optimal

iii) the homogeneity score increases as optimal k value is reached

V. Comparative Study of Performance of five Models and Conclusion

Model	**Performance:- accuracy score (in %)**
CNN	61.27
MLP	24.96
GNB	22.6
kNN	42.4
Minibatch K-Means	30

By analysing the performance of these algorithms on FER2013 dataset, we conclude a convolutional neural network is the best for predicting and classifying the images according to their emotions. The individual algorithms are fine-tuned for performance using different hyperparameters and performance metrics.

Comparing the performance of MLP and CNN we see that MLP, even with accuracy boosting techniques applied, yields a much lower accuracy than CNN. This may be because of the fully connected nature of MLP wherein the number of parameters becomes unmanageable leading to redundancy and inefficiency. MLPs are also not translation invariant i.e. different angles of the same headshot may lead the model to believe that only one portion of the image may contain the eyes or certain intensity features. On the contrary, CNN considers the spatial correlation between the pixels and since layers are sparsely connected CNN architecture can be much deeper. Kernels and convolutional layers make CNN much easier to train and more efficient with image data.

Supervised learning algorithms like kNN and GNB yield poorer results as images of FER2013 dataset belong to multiple classes, are complex as there may be overlap between the images and may lead to inefficiency as it is high-dimensional. KNN performs much better than even the unsupervised clustering (k-means). This may be because kNN

is suitable for multi-class classification and the images of the test dataset are close to some instances in the training dataset hence kNN is able to compare and assign the correct label. K-means showed that we needed more clusters than the original seven. We conclude that k-means is not suitable for FER dataset as it predicts the optimal number of clusters as 256 using the pixels however there is more complex correlation between the images which can only be found using DL models. We also try to reduce the dimensionality of given FER2013 dataset using PCA and use the same for k-means clustering. The resultant PCA plot also showed k-means would not be able to find differentiable clusters in the FER dataset.

For future work, looking into improving CNNs for FER and trying to stack classifiers may yield better results.

CONTRIBUTIONS

UK worked on the implementation and analysis of CNN model, SupS implemented MLP algorithm, SubS implemented Naïve Bayes, its variations (GNB and BNB) and PCA, and VA implemented kNN and minibatch K-means algorithm. The contributors then wrote about their specific model implementation and gave an overall behind the working and performance analysis of individual models. EDA and related tasks were performed and compiled by UK and SubS. VA contributed the comparative analysis of implemented model performances and concluding remarks.

ACKNOWLEDGMENT

We thank our machine learning professor, Dr. S. Karthika, for helping us learn the concepts behind these models we chose to explore and for providing a solid foundation in understanding ML and DL models' working, as well as a chance to work on this paper. We thank the department of IT in SSN College of Engineering for providing a platform to showcase what we learnt with this term paper.

REFERENCES

[1] Yousif Khaireddin, Zhuofa Chen, *Facial Emotion Recognition: State of the Art Performance on FER2013*, 2021, https://arxiv.org/abs/2105.03588

[2] Imen Tayari Meftah, Nhan Le Thanh & Chokri Ben Amar, *Emotion Recognition Using KNN Classification for User Modeling and Sharing of Affect States* in the Lecture Notes in Computer Science book series (LNIP,volume 4418), ICONIP 2012: "Neural Information Processing", pp 234–242, https://link.springer.com/chapter/10.1007/978-3-642-34475-6_29#citeas

[3] Abu Sayeed Md. Sohail & Prabir Bhattacharya, *Classification of Facial Expressions Using K-Nearest Neighbor Classifier* in MIRAGE 2007: "Computer Vision/Computer Graphics Collaboration Techniques", pp 555–566.

[4] Swapna Subudhiray, Hemanta Kumar Palo, Niva Das, "K-nearest neighbor based facial emotion recognition using effective features" in IAES International Journal of Artificial Intelligence (IJ-AI), Vol. 12, No. 1, March 2023, pp. 57~65 ISSN: 2252-8938, DOI: 10.11591/ijai.v12.i1.pp57-65.

[5] Minh-An Quinn, Grant Sivesind, Guilherme Reis, *Real-time Emotion Recognition From Facial Expressions*, 2017, CS229 Stanford University, http://cs229.stanford.edu/proj2017/final-reports/5243420.pdf.

[6] Avigyan Sinha, Aneesh R P, "Real Time Facial Emotion Recognition using Deep Learning," *IJIIE - International Journal of Innovations & Implementations in Engineering*, ISSN 2454- 3489, 2019, December Edition Volume 1, http://ijiie.org/download/IJIIE_2019DEC1001VOL1.pdf

[7] Goodfellow, Ian J., et al. "Challenges in representation learning: A report on three machine learning contests." International Conference on Neural Information Processing. Springer, Berlin, Heidelberg, 2013.