Fruits 360

Soumik Roy, Tanisha Jain
B20AI042, B20CS093

Abstract : This report presents our observations and analysis while developing a classification model for Fruits 360 Classification. We applied image preprocessing on the provided data, trained different ML models, did hyperparameter tuning and tested them. We also compared our different approaches for this problem. Finally, to have a real life working application of our work done in this machine learning problem of Fruits-360, we developed a fully functional android application from scratch which classifies given images into possible fruits.

I. Introduction

Image classification is the task of assigning a label to an image from a predefined set of categories. In a supervised image classification problem, a set of target classes(objects in images) to be identified is defined and we train a model to recognize them using labeled example photos.
In this project, we classify the images of fruits and vegetables from the Fruit 360 dataset.

About Dataset

Fruit 360 dataset contains 90380 images of 131 fruits and vegetables.
Images are 100 pixel by 100 pixel and are color (RGB) images.
There are 67,692 images in the training dataset and 22,688 images in the test dataset.

II. Methodology

Overview

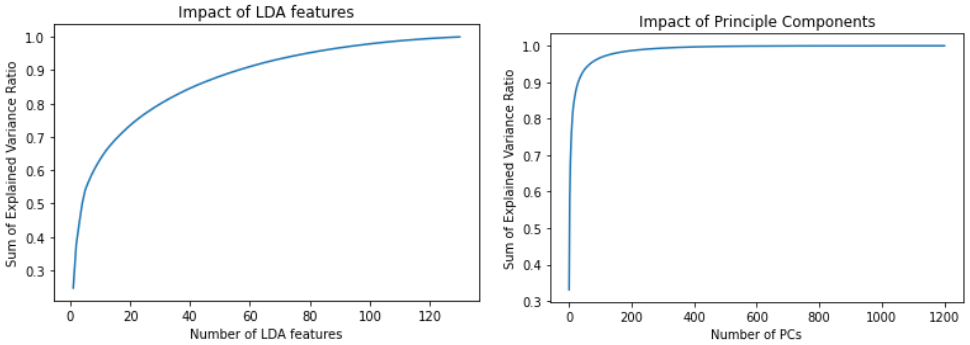
There are various classification algorithms present out of which we implemented the following:

- K-Nearest Neighbors(with and without LDA, PCA)
- Random Forest Classifier(with and without LDA, PCA)
- Convolution Neural Network

Data Preprocessing

- **Loading the dataset**
  Firstly, all the image files were imported into the notebook using the Kaggle API call.Then each image from each fruit folder was first scaled into 20x20 pixels and then a flattened version of its numerical array was concatenated into the respective dataset (test/train).

- **Dimensionality Reduction**
    1. **LDA:** Linear Discriminant Analysis(LDA) is a dimensionality reduction technique that focuses on finding a feature subspace which maximizes the separability among the known categories(classes) in the target variable. We created an LDA-applied version of the dataset.
    2. **PCA:** Principal Component Analysis is an unsupervised dimensionality reduction technique. It works by considering the variance of each attribute because the high attribute shows the good split between the classes, and hence it reduces the dimensionality. We created a PCA applied version of the dataset.

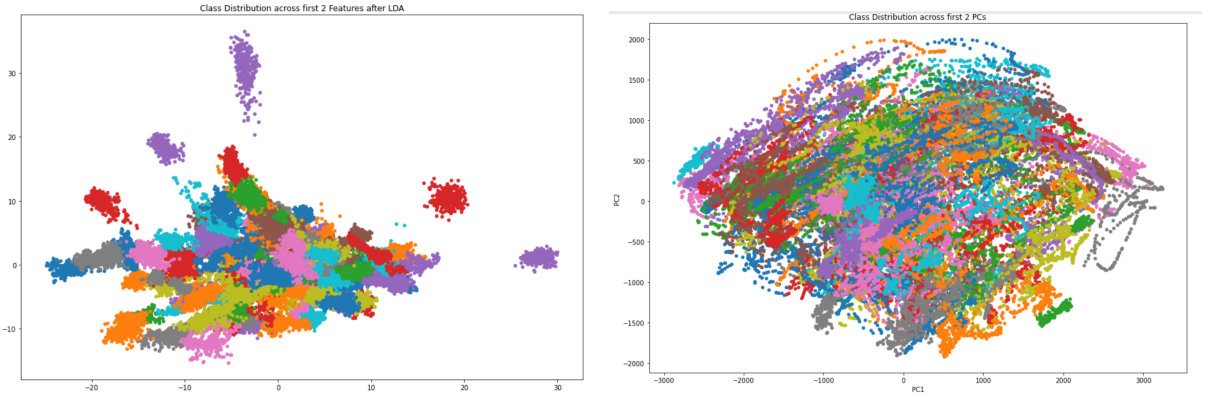


As we can see the Explained variance ratio sum for LDA reaches around 90% at around 120 features, while for PCA its just at around 30 features.

- **Creating Subsets containing only 10% data**

    We created a smaller subset of the original dataset having only 10% data for hyperparameter tuning. We did the same for all 3, the original data, and LDA/PCA applied data.

- **Visualization**



From the above plots, we expect LDA to perform better than PCA (better seperability), which we'll explore further in the report.

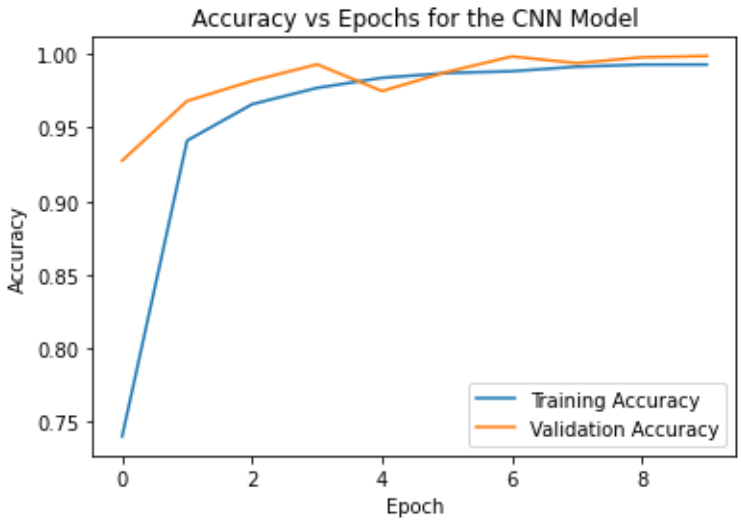
Implementation of Classification Algorithms

- **K-Nearest Neighbours:**
  KNN is a supervised learning algorithm used for classification and regression. It considers K nearest neighbours(data points) to predict the class in case of a classification problem.
  Following KNN classifiers were made:
    - ❖ KNN
    - ❖ KNN with LDA
    - ❖ KNN with PCA

  Hyperparameter tuning:
  We varied the number of neighbours from 3 to 10 and trained the model on the 10% dataset. The best nearest neighbour obtained was 3.

- **Random Forest Classifier:**
  Random forest is a supervised learning algorithm based upon ensemble learning techniques which combines many decision trees classifiers to provide aggregated results. It eradicates the limitations of a decision tree algorithm, reduces the overfitting of datasets and increases precision.
  Following Random Forest Classifiers were made:
    - ❖ Random Forest
    - ❖ Random Forest with LDA
    - ❖ Random Forest with PCA

- **Convolution Neural Network**
  Convolution Neural Network(CNN) is a class of neural networks that specializes in processing data that has a grid-like topology such as an image. It typically has three layers: a convolutional layer, a pooling layer, and a fully connected layer. The specifications of the CNN we built :
    - ❖ 2 Hidden Layers (ReLU activation) : Sizes = 140, 150 respectively
    - ❖ Output Layer (Softmax activation)
    - ❖ 10 epochs



III. Evaluation of models

Performance of the models implemented in different versions of the dataset are as follows :

PERFORMANCE ON VALIDATION DATASET

Model Used	Original Data	With PCA	With LDA
KNN	99.78	99.87	99.85
Random Forest	99.95	99.92	99.92
CNN	99.86	-	-

Metric used : Accuracy Score (%)

We see a better performance of PCA compared to LDA. This is because LDA specifically tries to reduce the dimensions from 1200 features to just 131 features which leads to a huge loss of information.

FINAL PERFORMANCE ON TEST DATASET

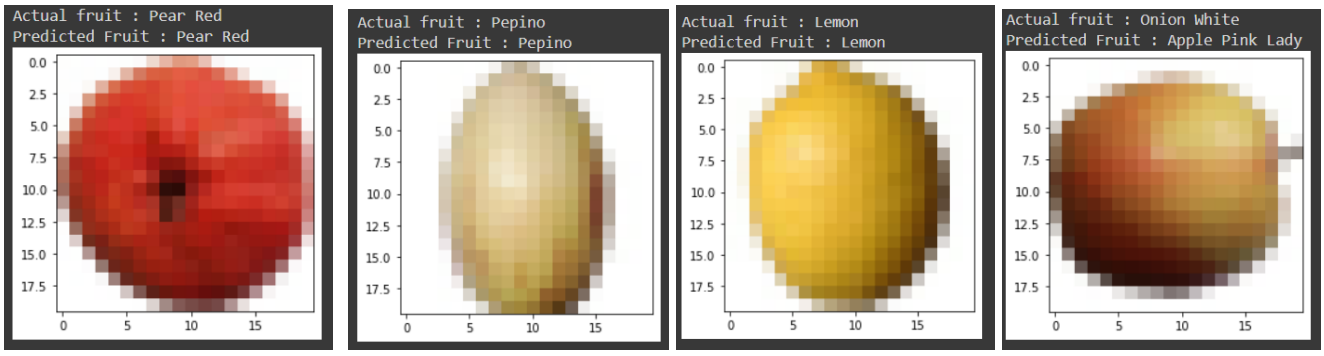
Model Used	Accuracy Score (in %)
KNN	89.86 (with LDA)
Random Forest	95.06 (original)
CNN	94.69

IV. Results and analysis

Finally as we can observe, Random Forest and CNN performed quite well on the data. However if we apply the two models in order to predict random images taken from the internet, CNN performs far better than Random Forest. This is because CNN does not ignore the fact that the positioning of the fruit in the image provided will not always be centered. CNN performs convolution on the given data and learns in a way that allows the model to make correct predictions even after slightly differently positioned fruit images are provided.

Hence we finalize CNN as the Classifier that best works on the Fruits-360 data. Now we saved the learned tensorflow model of the CNN for further usages.

Some Random predictions by CNN :



V. Pipeline

Now the task that remains is to create an End-to-End pipeline in order to achieve a way to get a trained model and predictions from raw data. A class was created inorder to implement the pipeline and the class provides methods to :

- **Fit** : Given raw training data, perform required preprocessing on it, and return a model trained on the data. From above analysis, the model that is being returned is a Convolutional Neural Network.
- **Predict** : Classify given test data and predict the fruit, an image might belong to. Returns predicted values for each row in test data.
- **Score** : Get the accuracy score of the model on making predictions on the given test data.

Attributes of the Pipeline class :

- Pipeline.model : The trained CNN model
- Pipeline.img_shape : The shape of the images using which given data was created.

VI. Application (bonus work)

We created an Android mobile application based on ML done in the project. It classifies any given image into a fruit (131 possible fruits). The application is built from scratch on Flutter and the ML model it uses is a Tensorflow CNN trained upon the Fruits-360 dataset.
You can find the apk for the application, and some images you can download and test the application in the following link :
https://drive.google.com/drive/folders/1B2w8nOeuNImNt-5URcl1vw6LdkePfTlp?usp=sharing

VII. Contributions

- ★ Soumik Roy (B20AI042): Loading and creating the dataset, KNN, Hyperparameter tuning, final models saved(Pickle and tensorflow-lite), Fruit-classifying App, Report
- ★ Tanisha Jain (B20CS093): Data Preprocessing, Visualization, Random Forest, CNN, Pipeline, Report

VIII. References

- https://towardsdatascience.com/introduction-to-convolutional-neural-networks-cnn-with-tensorflow-57e2f4837e18
- https://www.section.io/engineering-education/image-preprocessing-in-python/
- https://www.tensorflow.org/lite/guide

Github Repository

https://github.com/Soumik-Roy/Fruits-360