Savitribai Phule Pune University
Modern Education Society's College of Engineering, Pune
19, Bund Garden, V.K. Joag Path, Pune – 411001.

ACCREDITED BY NAAC WITH "A" GRADE (CGPA – 3.13)

DEPARTMENT OF COMPUTER ENGINEERING

A REPORT

ON

Fruit Classification using Support Vector Machine

B.E. (COMPUTER)

SUBMITTED BY

Mr. Yogen Ghodke (71818291L)
Mr. Sudesh Pawar (71818502B)
Miss. Shravani Kanade (71818357G)

UNDER THE GUIDANCE OF

Prof. Gopal B. Deshmukh

(Academic Year: 2020-2021)

Savitribai Phule Pune University
Modern Education Society's College of Engineering, Pune
19, Bund Garden, V.K. Joag Path, Pune – 411001.

ACCREDITED BY NAAC WITH "A" GRADE (CGPA – 3.13)

DEPARTMENT OF COMPUTER ENGINEERING

Certificate

This is to certify that project entitled

Fruit Classification using Support Vector Machine
has been completed by
Mr. Yogen Ghodke (PRN. 71818291L)
Mr. Sudesh Pawar (PRN. 71818502B)
Miss. Shravani Kanade (PRN. 71818357G)

of BE COMP I in the Semester - II of academic year 2020-2021 in partial fulfillment of the Fourth Year of Bachelor degree in "Computer Engineering" as prescribed by the Savitribai Phule Pune University.

Prof. Gopal B. Deshmukh
Project Guide

(Dr.(Mrs.) N. F. Shaikh)
H.O.D

Place: MESCOE, Pune.
Date: /05/2021

ACKNOWLEDGEMENT

It gives us great pleasure and satisfaction in presenting this mini project on "Fruit Classification using Support Vector Machine".

We would like to express my deep sense of gratitude towards all faculty for their support and advice during the development.

We have furthermore to thank Computer Department HOD **Dr.(Mrs.) N. F. Shaikh** *for her pearls of wisdom and our Project Guide* **Prof. Gopal B. Deshmukh** *to encourage us to go ahead and for continuous guidance. His incisive and objective guidance and timely advice encouraged us with a constant flow of energy to continue the work. Finally, we must say that no height is ever achieved without some sacrifices made at some end and it us here we owe our special debt to our parents and our friends for showing their generous love and care throughout the entire period.*

We would like to thank all those, who have directly or indirectly helped us for the completion of the work during this mini project.

Yogen Ghodke (71818291L)
Sudesh Pawar (71818502B)
Shravani Kanade (71818357G)
B.E. Computer

Contents

List of Figures

Abstract

Food processing Technologies play important roles in supporting Industrial Growth in F&B Industry. The effectiveness in food processing becomes emerging in the food industry. The automatic fruit classification system is becoming important for use in many food processing industries. A novel approach for classifying, using support vector machines (SVM) is presented in this mini project. Fruit classification based on their shape is proposed in this work. The system differentiates the fruit based on their shape. Fast Fourier Transform (FFT) is extracted and later used as input to the SVM-based identifier. The fruit parameters are compared and classified, the results of computer simulation show that this technique produces better accuracy than that of the existing technique that is based on Deep Learning Methods. The SVM also requires less training time than Deep Learning Methods.

Keywords-*Food processing, fruit image, Fast Fourier Transform (FFT), Support Vector Machines (SVMs)*

Chapter 1

INTRODUCTION

1.1 Support Vector Machine

Support vector machine is highly preferred by many as it produces significant accuracy with less computation power. Support Vector Machine, abbreviated as SVM can be used for both regression and classification tasks. But, it is widely used in classification objectives.

1.1.1 What is SVM?

The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.

To separate the two classes of data points, there are many possible hyperplanes that could be chosen. Our objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.

1.1.2 Hyperplanes and Support Vectors

Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. Also, the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3.

Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.

1.2 Motivation

Nowadays, conventional fruit shorting systems still used human manually in order to separate many kinds of fruit. However, the conventional fruit shorting by human system

has potential to be missclassify, because of tired, sick or not focus. Furthermore, Many of the miss classification fruit has occurred mostly due to the fatigue factor of the human. One of the solutions to the problem is by applying the automation in the fruit shorting system which capable of sorting the fruit correctly. The automatic sorting fruit system is one of the solutions to classify these fruits. This demand motivated us to develop a ML-powered classification system which uses SVM as its core algorithm.

Chapter 2

Optimization

2.1 Problem Statement

This project uses Python Sci-kit Learn module for accurately classifying fruits with the help of Support Vector Classifier present in the SVM class.

2.2 Objectives

1. To accurately classify fruits based on feature.
2. Perform better classification than manual methods.
3. Prove why SVC is better than other classifiers for this task.
4. Reduce Manual Errors in identification.
5. Use Python Tools available to develop a robust system.
6. Emphasis on avoiding biased treatment towards any fruit i.e. Avoid Overfitting .

Chapter 3

PROJECT REQUIREMENTS

3.1 Packages Used

1. Python Imaging Library (PIL)
2. Sci-kit Learn
3. Seaborn
4. ZipFile
5. iPython
6. Miniconda

3.2 Dataset

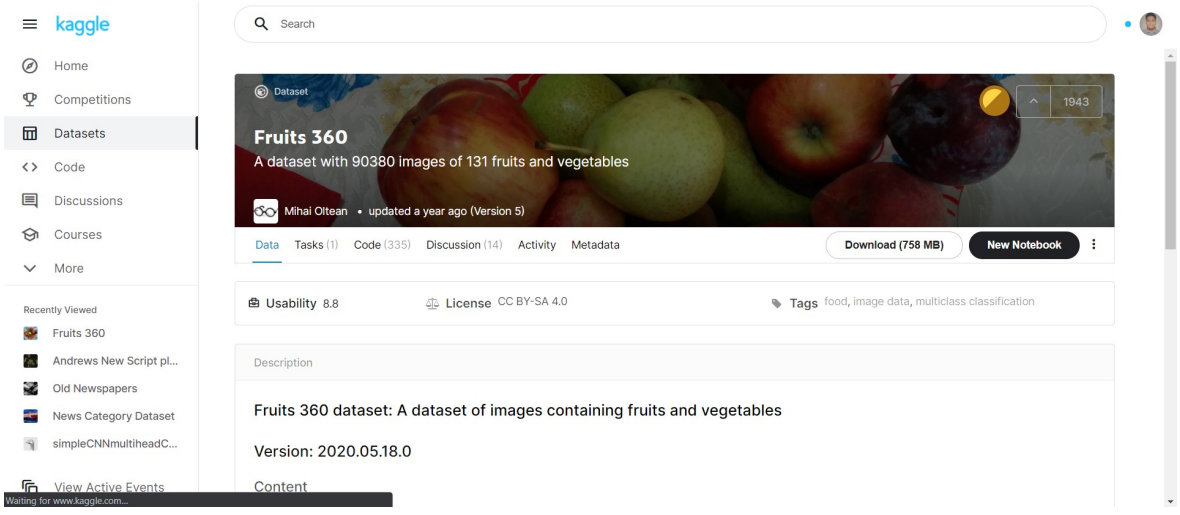


Figure 3.1: Kaggle Page

A dataset containing 90000 images of 120 Fruits, many of which are mostly concentrated in North American Region. For our project, we have taken a smaller subset of this dataset consisting of approx 12000 Images and 13 Fruits (Classes).
These Fruits are not necessarily very well defined in the images and therefore we use Python Pre-processing for further enhancement on these articles to create our final

dataset for usage.

3.2.1 Dataset Details

1. No. of Images : 90000
2. Size : 870 MB
3. No. of Classes: 120
4. Site of Origin: Kaggle
5. Concentrated on: North American Region

3.2.2 Technical Workflow

1. Images are loaded into the system
2. Image is converted to Grayscale.
3. Grayscale Image is serialized into an array.
4. Similar Preprocessing applied on Test data too.
5. SVC fitted on the training dataset.
6. Model tested and validated using the respective data.

3.3 Tech Stack

3.3.1 Pillow

Figure 3.2: Pillow

The Python Imaging Library adds image processing capabilities to your Python interpreter. This library provides extensive file format support, an efficient internal representation, and fairly powerful image processing capabilities.

The core image library is designed for fast access to data stored in a few basic pixel formats. It should provide a solid foundation for a general image processing tool.

3.3.2 Scikit-Learn

Figure 3.3: Scikit-Learn

Scikit-learn (formerly known as scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Scikit-learn is largely written in Python, and uses NumPy extensively for linear algebra and array operations. Furthermore, some core algorithms are written in Cython to improve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; In such cases, extending these methods with Python may not be possible.

Scikit-learn integrates well with many other Python libraries, such as Matplotlib & plotly for plotting, NumPy for array vectorization, Pandas dataframes, SciPy, and many more.

3.3.3 Seaborn

Figure 3.4: Seaborn

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

Seaborn helps you explore and understand your data. Its plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots. Its dataset-oriented, declarative API lets you focus on what the different elements of your plots mean, rather than on the details of how to draw them.

3.3.4 Zipfile

Figure 3.5: Zipfile

The ZIP file format is a common archive and compression standard. This module provides tools to create, read, write, append, and list a ZIP file. Any advanced use of this module will require an understanding of the format.

This module does not currently handle multi-disk ZIP files. It can handle ZIP files that use the ZIP64 extensions. It supports decryption of encrypted files in ZIP archives, but it currently cannot create an encrypted file. Decryption is extremely slow as it is implemented in native Python rather than C.

Chapter 4

IMPLEMENTATION OF PROJECT

The following pages contain the entire Implementation in form of Jupyter Notebook.

In [1]:

```
import pandas as pd
import numpy as np
import cv2
import os
import matplotlib.pyplot as plt
from sklearn.utils import shuffle

os.listdir('fruits-360/Training')[0:13]
```

Out[1]:

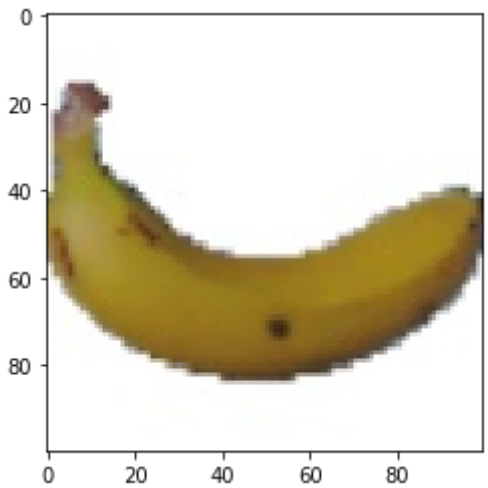
```
['Apple',
 'Banana',
 'Blueberry',
 'Cherry',
 'Grapes',
 'Lychee',
 'Mango',
 'Orange',
 'Papaya',
 'Pineapple',
 'Pomegranate',
 'Strawberry',
 'Watermelon']
```

In [2]:

```
path = 'fruits-360/Training/Banana/71_100.jpg'
im = cv2.imread(path)
b,g,r = cv2.split(im)
im2 = cv2.merge([r,g,b])
plt.imshow(im2)
```

Out[2]:

```
<matplotlib.image.AxesImage at 0x2c265bc6448>
```

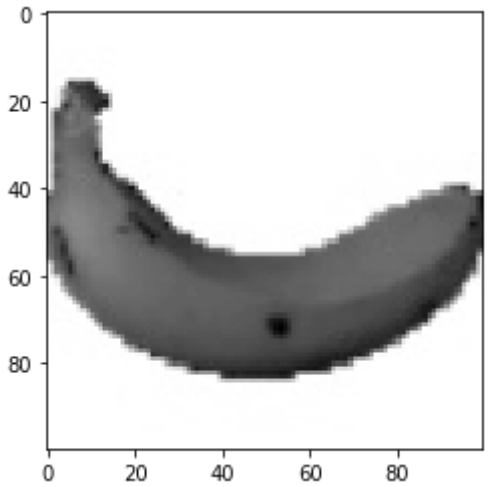


In [3]:

```
grayim = cv2.cvtColor(im2,cv2.COLOR_RGB2GRAY)
plt.imshow(grayim,cmap = 'gray')
```

Out[3]:

```
<matplotlib.image.AxesImage at 0x2c265c71908>
```



In [4]:

```
grayimg = grayim.astype('float')/255
grayimg = grayimg.flatten()
grayimg = pd.Series(grayimg)
grayimg
```

Out[4]:

```
0       1.0
1       1.0
2       1.0
3       1.0
4       1.0
       ...
9995    1.0
9996    1.0
9997    1.0
9998    1.0
9999    1.0
Length: 10000, dtype: float64
```

In [5]:

```
path = 'fruits-360/Training/'
cols = np.arange(grayimg.shape[0])
df = pd.DataFrame(columns = cols)
labelcol = []
fruitlist = os.listdir(path)
x = 0

for f in fruitlist[0:13] :
    fruitpath = '%s%s' % (path,f)
    imagelist = os.listdir(fruitpath)
    for i in imagelist:
        imagepath = '%s/%s' % (fruitpath,i)
        image = cv2.imread(imagepath)
        b,g,r = cv2.split(image)
        image = cv2.merge([r,g,b])
        imagegray = cv2.cvtColor(image,cv2.COLOR_RGB2GRAY)
        imagegray = imagegray.astype('float')/255
        imagegray = imagegray.flatten()
        df.loc[x] = imagegray
        x = df.shape[0] + 1
        labelcol.append(f)
```

In [6]:

```
df['label'] = labelcol
df
```

Out[6]:

	0	1	2	3	4	5	6	7	8	9
0	0.996078	0.996078	1.000000	1.0	1.0	1.000000	1.000000	0.996078	0.996078	0.996078
2	0.996078	0.996078	1.000000	1.0	1.0	0.996078	0.996078	0.996078	0.996078	0.996078
3	0.992157	0.996078	0.996078	1.0	1.0	0.996078	0.996078	0.996078	0.992157	0.992157
4	0.992157	0.996078	0.996078	1.0	1.0	0.996078	0.996078	0.996078	0.992157	0.992157
5	0.996078	0.996078	0.996078	1.0	1.0	1.000000	1.000000	0.996078	0.996078	0.996078
...	...	...	...	...	...	...	...	...	...	...
6322	1.000000	1.000000	1.000000	1.0	1.0	1.000000	1.000000	1.000000	1.000000	1.000000
6323	1.000000	1.000000	1.000000	1.0	1.0	1.000000	1.000000	1.000000	1.000000	1.000000
6324	1.000000	1.000000	1.000000	1.0	1.0	1.000000	1.000000	1.000000	1.000000	1.000000
6325	1.000000	1.000000	1.000000	1.0	1.0	1.000000	1.000000	1.000000	1.000000	1.000000
6326	1.000000	1.000000	1.000000	1.0	1.0	1.000000	1.000000	1.000000	1.000000	1.000000

6326 rows × 10001 columns

In [7]:

```
df['label'].value_counts(normalize = True)
```

Out[7]:

```
Papaya          0.077774
Apple           0.077774
Pomegranate     0.077774
Strawberry      0.077774
Cherry          0.077774
Mango           0.077458
Banana          0.077458
Pineapple       0.077458
Grapes          0.077458
Lychee          0.077458
Orange          0.075719
Watermelon      0.075087
Blueberry       0.073032
Name: label, dtype: float64
```

In [8]:

```
df = shuffle(df).reset_index(drop = True)
df
```

Out[8]:

	0	1	2	3	4	5	6	7	8
0	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
1	0.996078	0.996078	0.996078	0.996078	0.996078	0.996078	0.996078	0.996078	0.996078
2	0.996078	0.996078	0.996078	1.000000	1.000000	1.000000	0.996078	0.996078	0.996078
3	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
4	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
...	...	...	...	...	...	...	...	...	..
6321	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
6322	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
6323	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
6324	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
6325	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

6326 rows × 10001 columns

In [9]:

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix,classification_report

X = df.drop('label',axis = 1)
y = df['label']

X_train,X_test,y_train,y_test = train_test_split(X,y,random_state = 0,stratify = y)
```

In [10]:

```
from sklearn.svm import SVC
svm_model = SVC().fit(X_train,y_train)
trainscore = svm_model.score(X_train,y_train)
testscore = svm_model.score(X_test,y_test)
y_pred = svm_model.predict(X_test)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWa
rning: The default value of gamma will change from 'auto' to 'scale' in vers
ion 0.22 to account better for unscaled features. Set gamma explicitly to 'a
uto' or 'scale' to avoid this warning.
  "avoid this warning.", FutureWarning)
```

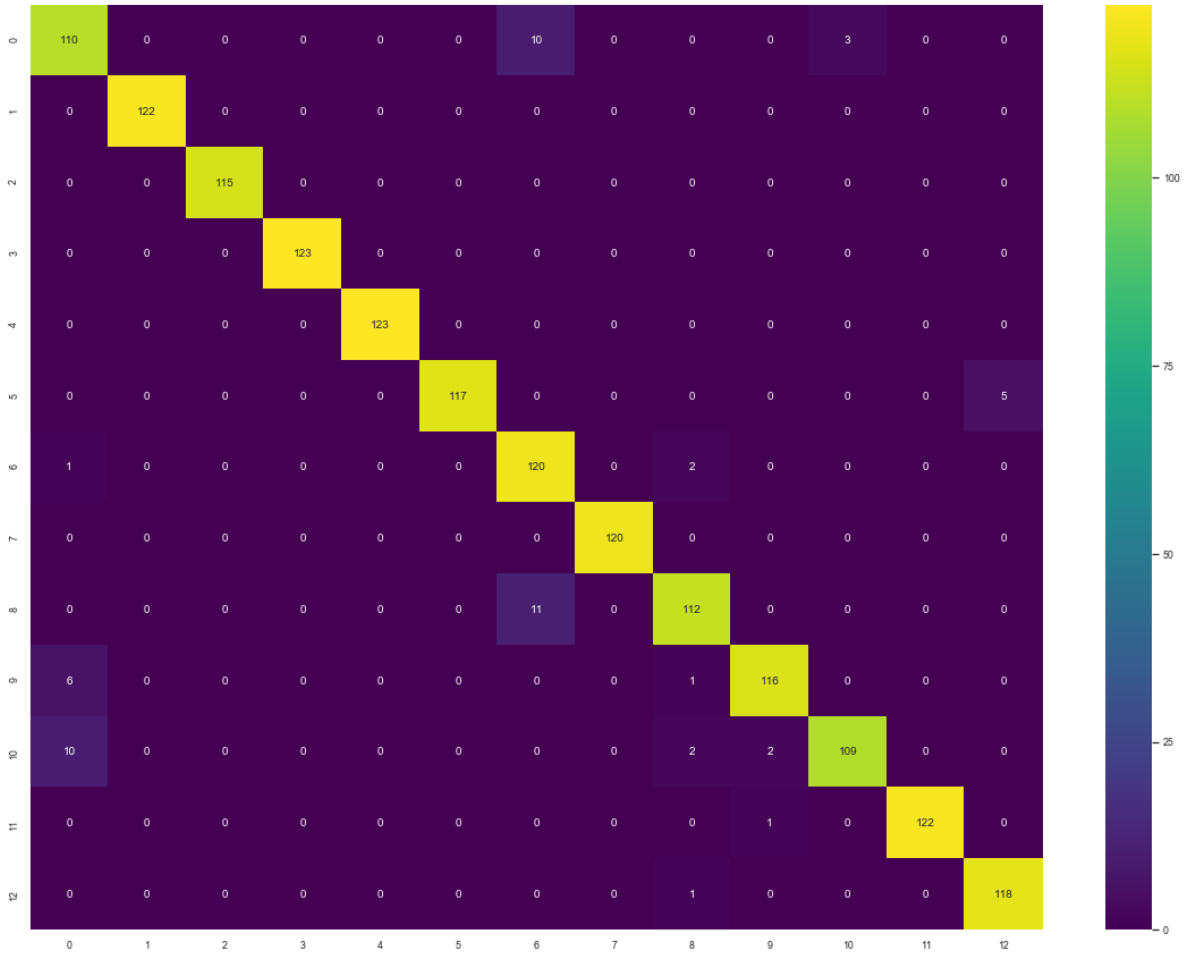
In [58]:

```
import seaborn as sns
print("CLASSIFICATION REPORT FOR SVM")
print("Confusion Matrix:")
cf_matrix = confusion_matrix(y_test,y_pred)
sns.set(font_scale=0.8)
plt.figure(figsize = (20,15))
sns.heatmap(cf_matrix, annot=True,fmt='d', annot_kws={"size": 10},cmap="viridis").set_ylim(
```

```
CLASSIFICATION REPORT FOR SVM
Confusion Matrix:
```

Out[58]:

```
(13, 0)
```



In [42]:

```
print(classification_report(y_test,y_pred))
print("Train Score:",trainscore)
print("Test Score",testscore)
```

```
              precision    recall  f1-score   support

       Apple       0.87      0.89      0.88       123
      Banana       1.00      1.00      1.00       122
   Blueberry       1.00      1.00      1.00       115
      Cherry       1.00      1.00      1.00       123
      Grapes       1.00      1.00      1.00       123
      Lychee       1.00      0.96      0.98       122
       Mango       0.85      0.98      0.91       123
      Orange       1.00      1.00      1.00       120
      Papaya       0.95      0.91      0.93       123
   Pineapple       0.97      0.94      0.96       123
 Pomegranate       0.97      0.89      0.93       123
  Strawberry       1.00      0.99      1.00       123
  Watermelon       0.96      0.99      0.98       119

    accuracy                           0.97      1582
   macro avg       0.97      0.97      0.97      1582
weighted avg       0.97      0.97      0.97      1582

Train Score: 0.9690134907251264
Test Score 0.9652338811630847
```

In [ ]:

```

```

Chapter 5

CONCLUSION

The automatic fruit classification system has been proposed and developed as part of the mini project. This technique compared with the existing method based on deep learning with help of parameters such as Recall, Precision, Confusion Matrix etc. The accuracy of the SVM based fruit classification system was found better than to the existing method based on ANN/CNN. It was also clear, from our computer simulation results, that the proposed method requires much less training time than the ones trained using deep learning methods. These results are two different fruits in the system considered.