Experiment:1.1

Aim: Write a program to implement various feature extraction techniques for image classification.

Software Required: Any Python IDE (e.g.: PyCharm, Jupyter Notebook, GoogleCollab)

Technique used:

Various feature extraction methods are often employed in picture classification problems. These methods seek to extract pertinent data from photos and turn it into meaningful representations that machine learning algorithms may use to classify images. Feature extraction methods include:

1. Histogram of Oriented Gradients (HOG): The Histogram of Oriented Gradients (HOG) is a popular feature descriptor technique in computer vision and image processing. It analyses the distribution of edge orientations within an object to describe its shape and appearance. The HOG method involves computing the gradient magnitude and orientation for each pixel in an image and then dividing the image into small cells.

Pseudo code/Algorithms/Flowchart/Steps:

1. Import Necessary libraries.
2. Load the image.
3. Resize the image.
4. Convert image from BRG to RGB.
5. Printing resized image shape.
6. Calculating pixels and displaying image.

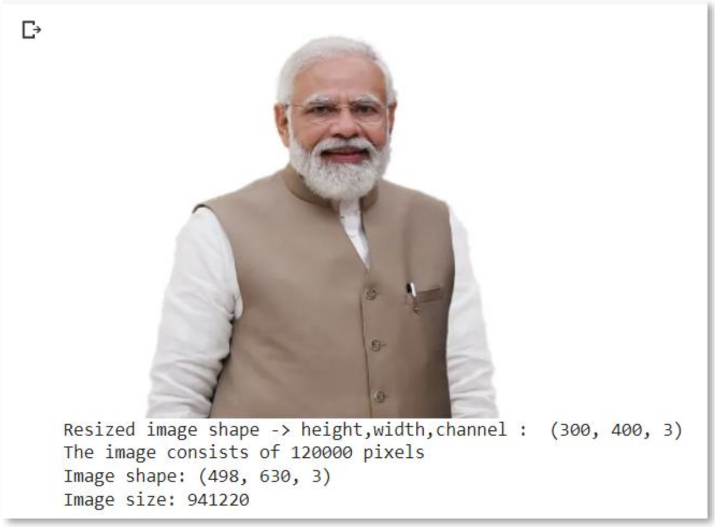
Implementation:

```
import numpy as np
from google.colab.patches import cv2_imshow
import cv2
import matplotlib.pyplot as plt

# reading images from folder using imread
image = cv2.imread('/content/drive/MyDrive/Colab Notebooks/Images/Modi-Ji.webp')
```

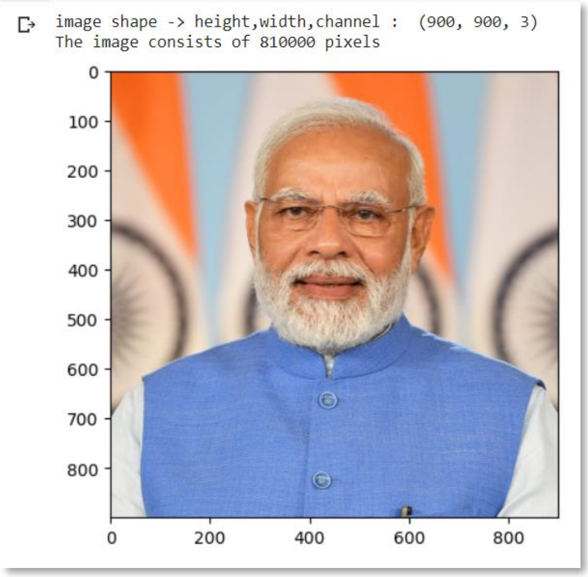
```
# Resize the image
resized_image = cv2.resize(image, (400, 300))

# Display the image using cv2_imshow
cv2_imshow(resized_image)

# resized_image.shape has (height, width, channels)
# for a RGB image, the channel would be 3 (one for each of red, green, and blue),
# while grayscale images would have just one channel.
print("Resized image shape -> height,width,channel : ",resized_image.shape)

# calculating and printing the total number of pixels in the image by multiplying
its height and width.
print("The image consists of %i pixels" % (resized_image.shape[0] *
resized_image.shape[1]))
resized_image = cv2.cvtColor(image,cv2.COLOR_BGR2RGB)
plt.axis('off')

# display the image
plt.imshow(resized_image);

print("Image shape:", resized_image.shape)

# product of dimensions (height, width, and channels) of image
print("Image size:", resized_image.size)
```

```
import matplotlib.pyplot as plt
# using the imread() function to read an image file
image=plt.imread('/content/drive/MyDrive/Colab Notebooks/Images/Narendra Modi.jpg')
# image.shape has (height, width, channels)
# for a RGB image, the channel would be 3 (one for each of red, green, and blue),
# while grayscale images would have just one channel.
print("image shape -> height,width,channel : ",image.shape)
# calculating and printing the total number of pixels in the image by multiplying
its height and width.
print("The image consists of %i pixels" % (image.shape[0] * image.shape[1]))
# display the image
plt.imshow(image);
```

Output Screenshots:



```
Resized image shape -> height,width,channel :  (300, 400, 3)
The image consists of 120000 pixels
Image shape: (498, 630, 3)
Image size: 941220
```



```
image shape -> height,width,channel :  (900, 900, 3)
The image consists of 810000 pixels
```

HOG Features using skimage Library:

Pseudo code/Algorithms/Flowchart/Steps:

1. Import Necessary libraries.
2. Load/Read the image.
3. Resizing the image and printing resized image shape.
4. Creating and visualizing HOG Features.
5. Printing dimension of HOG feature vector.

Implementation:

```
# Importing Libraries
from skimage.io import imread
from skimage.transform import resize
from skimage.feature import hog
import matplotlib.pyplot as plt
```

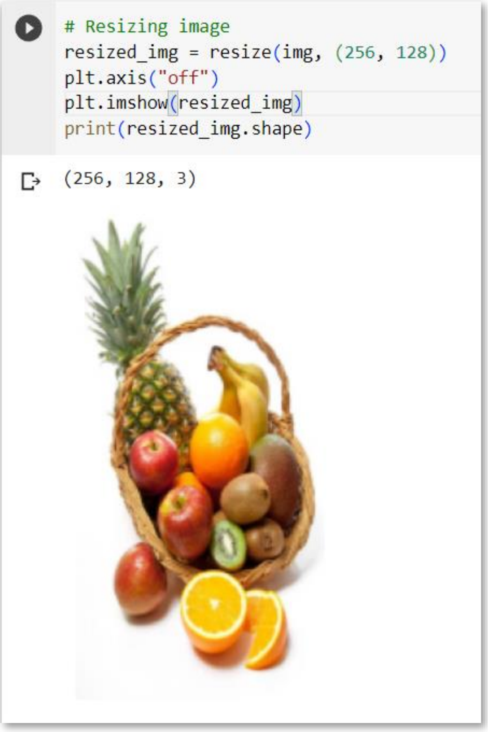
```
# Reading the image
img = imread('/content/drive/MyDrive/Colab Notebooks/Images/fruit.webp')
plt.axis("off")
plt.imshow(img)
print(img.shape)
```

```
# Resizing image
resized_img = resize(img, (256, 128))
plt.imshow(resized_img)
print(resized_img.shape)
```

```
# Creating and visualizing HOG Features
fd, hog_image = hog(resized_img, orientations=9, pixels_per_cell=(16, 16),
                    cells_per_block=(2, 2), visualize=True, multichannel=True)
plt.imshow(hog_image, cmap="gray")
plt.show()
```

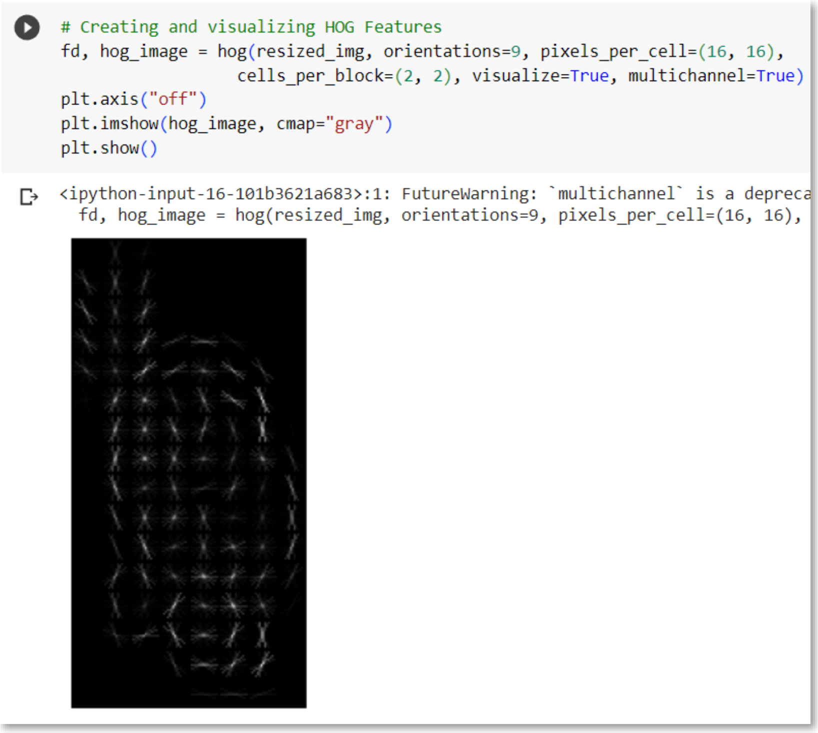
```
#  shape of the feature matrix or dimension of HOG feature vector
fd.shape
```

Output Screenshots:



```
# Resizing image
resized_img = resize(img, (256, 128))
plt.axis("off")
plt.imshow(resized_img)
print(resized_img.shape)
```

```
(256, 128, 3)
```



```
# Creating and visualizing HOG Features
fd, hog_image = hog(resized_img, orientations=9, pixels_per_cell=(16, 16),
                    cells_per_block=(2, 2), visualize=True, multichannel=True)
plt.axis("off")
plt.imshow(hog_image, cmap="gray")
plt.show()
```

```
<ipython-input-16-101b3621a683>:1: FutureWarning: `multichannel` is a depreca
  fd, hog_image = hog(resized_img, orientations=9, pixels_per_cell=(16, 16),
```