INTRODUCTION

NAME: DHRUV DESAI
COLLEGE: CHAROTAR UNIVERSITY OF TECHNOLOGY(CSPIT)
BRANCH: COMPUTER ENGINEERING
YEAR: SECOND

Content-Major_project_1_Report

Aim: Choose any dataset of your choice and apply a suitable classifier/regressor and if possible, deploy it on Heroku.

Code:

```
#importing lib
import numpy as np
from sklearn.linear_model import LinearRegression
```

```
[107] # importing dataset
      import pandas as pd
      df = pd.read_csv('/content/Video_Games_Sales_as_at_22_Dec_2016.csv')
      dx=df.dropna()
      dx
```

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Score	User_Count	Developer	Rating
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	8	322.0	Nintendo	E
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.68	12.76	3.79	3.29	35.52	82.0	73.0	8.3	709.0	Nintendo	E
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.61	10.93	3.28	2.95	32.77	80.0	73.0	8	192.0	Nintendo	E
6	New Super Mario Bros.	DS	2006.0	Platform	Nintendo	11.28	9.14	6.50	2.88	29.80	89.0	65.0	8.5	431.0	Nintendo	E
7	Wii Play	Wii	2006.0	Misc	Nintendo	13.96	9.18	2.93	2.84	28.92	58.0	41.0	6.6	129.0	Nintendo	E
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
16667	E.T. The Extra-Terrestrial	GBA	2001.0	Action	NewKidCo	0.01	0.00	0.00	0.00	0.01	46.0	4.0	2.4	21.0	Fluid Studios	E
16677	Mortal Kombat: Deadly Alliance	GBA	2002.0	Fighting	Midway Games	0.01	0.00	0.00	0.00	0.01	81.0	12.0	8.8	9.0	Criterion Games	M
16696	Metal Gear Solid V: Ground Zeroes	PC	2014.0	Action	Konami Digital Entertainment	0.00	0.01	0.00	0.00	0.01	80.0	20.0	7.6	412.0	Kojima Productions	M
16700	Breach	PC	2011.0	Shooter	Destineer	0.01	0.00	0.00	0.00	0.01	61.0	12.0	5.8	43.0	Atomic Games	T
16706	STORM: Frontline Nation	PC	2011.0	Strategy	Unknown	0.00	0.01	0.00	0.00	0.01	60.0	12.0	7.2	13.0	SimBin	E10+

6825 rows × 16 columns

```
[130] dx.size
```

```
109200
```

```
[130] dx.size
```

```
109200
```

```
[131] dx.shape
```

```
(6825, 16)
```

```
#Separate the independent variable(s) (X) and dependent variable (y) from the dataset.
X = df['Critic_Score'].values.reshape(-1,1)
y = df['Global_Sales'].values.reshape(-1,1)
dx
```

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Score	User_Count	Developer	Rating
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	8	322.0	Nintendo	E
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.68	12.76	3.79	3.29	35.52	82.0	73.0	8.3	709.0	Nintendo	E
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.61	10.93	3.28	2.95	32.77	80.0	73.0	8	192.0	Nintendo	E
6	New Super Mario Bros.	DS	2006.0	Platform	Nintendo	11.28	9.14	6.50	2.88	29.80	89.0	65.0	8.5	431.0	Nintendo	E
7	Wii Play	Wii	2006.0	Misc	Nintendo	13.96	9.18	2.93	2.84	28.92	58.0	41.0	6.6	129.0	Nintendo	E
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
16667	E.T. The Extra-Terrestrial	GBA	2001.0	Action	NewKidCo	0.01	0.00	0.00	0.00	0.01	46.0	4.0	2.4	21.0	Fluid Studios	E
16677	Mortal Kombat: Deadly Alliance	GBA	2002.0	Fighting	Midway Games	0.01	0.00	0.00	0.00	0.01	81.0	12.0	8.8	9.0	Criterion Games	M
16696	Metal Gear Solid V: Ground Zeroes	PC	2014.0	Action	Konami Digital Entertainment	0.00	0.01	0.00	0.00	0.01	80.0	20.0	7.6	412.0	Kojima Productions	M
16700	Breach	PC	2011.0	Shooter	Destineer	0.01	0.00	0.00	0.00	0.01	61.0	12.0	5.8	43.0	Atomic Games	T
16706	STORM: Frontline Nation	PC	2011.0	Strategy	Unknown	0.00	0.01	0.00	0.00	0.01	60.0	12.0	7.2	13.0	SimBin	E10+

6825 rows × 16 columns

```
[127] #Create an instance of the LinearRegression class from scikit-learn and fit it to the data
      from sklearn.linear_model import LinearRegression as lr
      model = lr()
```

```
[128]
      # the trained model to make predictions on new data.
      lr = LinearRegression()
      lr.fit(X, y)
```

```
      ▾ LinearRegression
      LinearRegression()
```

```
[125] #the trained model to make predictions on new data.y_pred = lr.predict(X)
      y_pred
```

```
      array([[0.97023057],
             [1.17202224],
             [1.10475835],
             ...,
             [1.10475835],
             [0.46575139],
             [0.43211944]])
```

```
      #Finally, evaluate the performance of the model by calculating metrics such as the R-squared value
      from sklearn.metrics import r2_score

      r2 = r2_score(y, y_pred)


      print(f"R-squared: {r2}")
```

```
      R-squared: 0.05643272118927545
```

Content-Major_project_2_Report

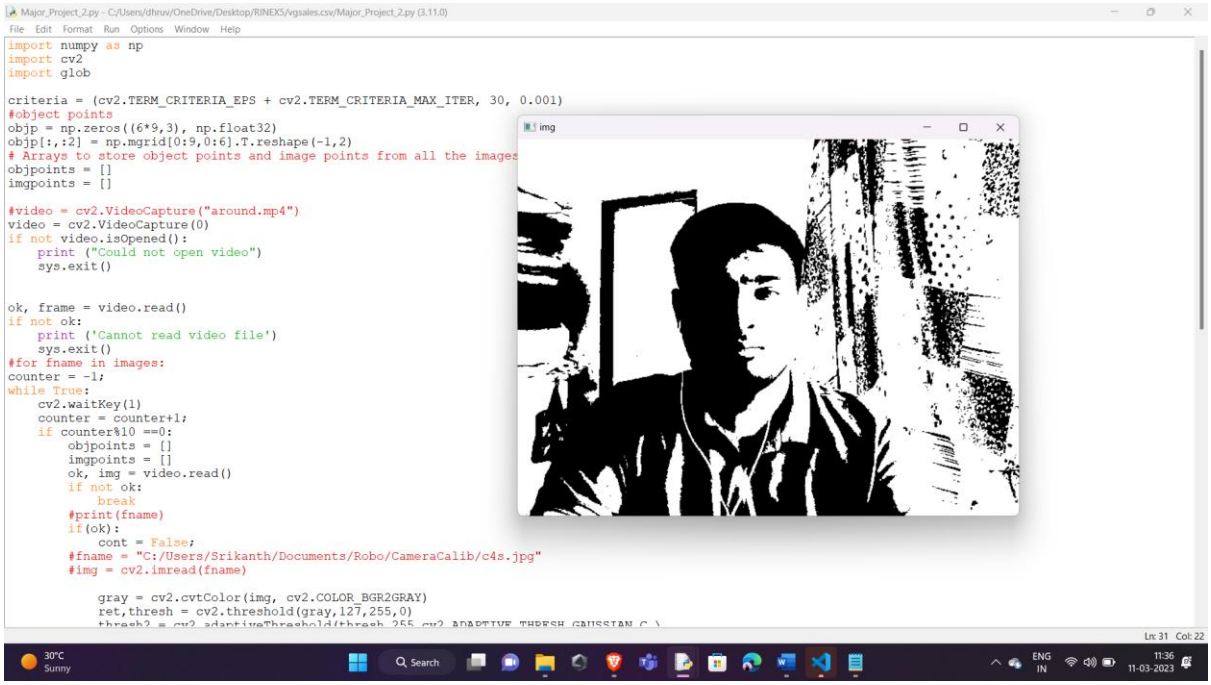
Aim: Create any of the Image Processing using NumPy and OpenCV.

Code:

```
import numpy as np
import cv2
import glob

criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 30, 0.001)
#object points
objp = np.zeros((6*9,3), np.float32)
objp[:,:2] = np.mgrid[0:9,0:6].T.reshape(-1,2)
# Arrays to store object points and image points from all the images.
objpoints = []
imgpoints = []

#video = cv2.VideoCapture("around.mp4")
video = cv2.VideoCapture(0)
if not video.isOpened():
   print ("Could not open video")
   sys.exit()


ok, frame = video.read()
if not ok:
   print ('Cannot read video file')
   sys.exit()
#for fname in images:
counter = -1;
while True:
   cv2.waitKey(1)
   counter = counter+1;
   if counter%10 ==0:
      objpoints = []
      imgpoints = []
      ok, img = video.read()
      if not ok:
         break
      #print(fname)
      if(ok):
         cont = False;
      #fname = "C:/Users/Srikanth/Documents/Robo/CameraCalib/c4s.jpg"
      #img = cv2.imread(fname)

         gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
         ret,thresh = cv2.threshold(gray,127,255,0)
         thresh2 = cv2.adaptiveThreshold(thresh,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,\
```

```
            cv2.THRESH_BINARY,5,10)
        blur = cv2.GaussianBlur(thresh,(5,5),0)
    # Find the chess board corners
        ret, corners = cv2.findChessboardCorners(thresh, (9,6),None)
    # If found, add object points, image points
        if ret == True:
            objpoints.append(objp)
            corners2=cv2.cornerSubPix(thresh,corners, (11,11), (-1,-1), criteria)
            imgpoints.append(corners)
       # Draw and display the corners
            cv2.drawChessboardCorners(thresh, (9,6), corners2, ret,)
            cv2.imshow('img', thresh)
            cv2.waitKey(10)

            ret, mtx, dist, rvecs, tvecs = cv2.calibrateCamera(objpoints, imgpoints, gray.shape[::-
1], None, None)

            rotVec = np.zeros((3, 3), np.float32)
            rotVec,_=cv2.Rodrigues(np.array(rvecs))



            rotf = -rotVec.T
            cameraPosition =  np.dot(rotf,np.array(tvecs))
            print(cameraPosition)

        else:
        #cv2.imshow('img', blur)
        #cv2.waitKey(5000)
            cv2.imshow('img', thresh)
       #break
            if cv2.waitKey(1) & 0xFF == ord('q'):
                break

cv2.destroyAllWindows()

#print(objpoints[0])
# ret, mtx, dist, rvecs, tvecs = cv2.calibrateCamera(objpoints, imgpoints, gray.shape[::-1],
None, None
# rotVec = np.zeros((3, 3), np.float32)
# rotVec,_=cv2.Rodrigues(np.array(rvecs))
# rotf = -rotVec.T
# cameraPosition =  np.dot(rotf,np.array(tvecs))
# print(cameraPosition)
```

OUTPUT:


Major_Project_2.py - C:/Users/dhruv/OneDrive/Desktop/RINEX5/vgsales.csv/Major_Project_2.py (3.11.0)
File Edit Format Run Options Window Help
import numpy as np
import cv2
import glob
criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 30, 0.001)
#object points
objp = np.zeros((6*9,3), np.float32)
objp[:,:2] = np.mgrid[0:9,0:6].T.reshape(-1,2)
# Arrays to store object points and image points from all the images
objpoints = []
imgpoints = []
#video = cv2.VideoCapture("around.mp4")
video = cv2.VideoCapture(0)
if not video.isOpened():
print ("Could not open video")
sys.exit()
ok, frame = video.read()
if not ok:
print ('Cannot read video file')
sys.exit()
#for fname in images:
counter = -1;
while True:
cv2.waitKey(1)
counter = counter+1;
if counter%10 ==0:
objpoints = []
imgpoints = []
ok, img = video.read()
if not ok:
break
#print(fname)
if(ok):
cont = False;
#fname = "C:/Users/Srikanth/Documents/Robo/CameraCalib/c4s.jpg"
#img = cv2.imread(fname)
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
ret,thresh = cv2.threshold(gray,127,255,0)
img
Ln: 31 Col: 22
30°C
Sunny
Search
ENG
IN
11:36
11-03-2023