ai-program-day-6

May 5, 2024

[3]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[4]:
```
df=pd.read_csv('Position Salary .csv')
```

[5]:
```
df
```

[5]:

	Position	Level	Salary
0	Busienss Analyst	1	45000
1	Junior Consultant	2	50000
2	Senior Consultant	3	60000
3	Manager	4	120000
4	Country Manager	5	140000
5	Region Manager	6	130000
6	Partnor	7	100000
7	Senior Partnor	8	90000
8	C Level	9	95000
9	CEO	10	85000

[6]:
```
x=df.iloc[:,1:2].values
print(x)
```

```
[[ 1]
 [ 2]
 [ 3]
 [ 4]
 [ 5]
 [ 6]
 [ 7]
 [ 8]
 [ 9]
 [10]]
```

[7]:
```
y=df.iloc[:,2].values
print(y)
```

```
[ 45000  50000  60000 120000 140000 130000 100000  90000  95000  85000]
```

[8]:
```
from sklearn.tree import DecisionTreeRegressor
dt=DecisionTreeRegressor()
dt.fit(x,y)
print(dt)
```

```
DecisionTreeRegressor()
```

[9]:
```
x_grid=np.arange(min(x),max(x),0.1)
x_grid
```

```
C:\Users\janam\AppData\Local\Temp\ipykernel_30080\3435800416.py:1:
DeprecationWarning: Conversion of an array with ndim > 0 to a scalar is
deprecated, and will error in future. Ensure you extract a single element from
your array before performing this operation. (Deprecated NumPy 1.25.)
  x_grid=np.arange(min(x),max(x),0.1)
```

[9]:
```
array([1. , 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2. , 2.1, 2.2,
       2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3. , 3.1, 3.2, 3.3, 3.4, 3.5,
       3.6, 3.7, 3.8, 3.9, 4. , 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8,
       4.9, 5. , 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6. , 6.1,
       6.2, 6.3, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9, 7. , 7.1, 7.2, 7.3, 7.4,
       7.5, 7.6, 7.7, 7.8, 7.9, 8. , 8.1, 8.2, 8.3, 8.4, 8.5, 8.6, 8.7,
       8.8, 8.9, 9. , 9.1, 9.2, 9.3, 9.4, 9.5, 9.6, 9.7, 9.8, 9.9])
```

[10]:
```
x_grid=x_grid.reshape((len(x_grid),1))
```

[11]:
```
x_grid
```

[11]:
```
array([[1. ],
       [1.1],
       [1.2],
       [1.3],
       [1.4],
       [1.5],
       [1.6],
       [1.7],
       [1.8],
       [1.9],
       [2. ],
       [2.1],
       [2.2],
       [2.3],
       [2.4],
       [2.5],
       [2.6],
       [2.7],
```

```
[2.8],
[2.9],
[3. ],
[3.1],
[3.2],
[3.3],
[3.4],
[3.5],
[3.6],
[3.7],
[3.8],
[3.9],
[4. ],
[4.1],
[4.2],
[4.3],
[4.4],
[4.5],
[4.6],
[4.7],
[4.8],
[4.9],
[5. ],
[5.1],
[5.2],
[5.3],
[5.4],
[5.5],
[5.6],
[5.7],
[5.8],
[5.9],
[6. ],
[6.1],
[6.2],
[6.3],
[6.4],
[6.5],
[6.6],
[6.7],
[6.8],
[6.9],
[7. ],
[7.1],
[7.2],
[7.3],
[7.4],
```

```
       [7.5],
       [7.6],
       [7.7],
       [7.8],
       [7.9],
       [8. ],
       [8.1],
       [8.2],
       [8.3],
       [8.4],
       [8.5],
       [8.6],
       [8.7],
       [8.8],
       [8.9],
       [9. ],
       [9.1],
       [9.2],
       [9.3],
       [9.4],
       [9.5],
       [9.6],
       [9.7],
       [9.8],
       [9.9]])
```

[12]:
```
dt.predict(x_grid)
```

[12]:
```
array([ 45000.,  45000.,  45000.,  45000.,  45000.,  45000.,  50000.,
        50000.,  50000.,  50000.,  50000.,  50000.,  50000.,  50000.,
        50000.,  50000.,  60000.,  60000.,  60000.,  60000.,  60000.,
        60000.,  60000.,  60000.,  60000.,  60000., 120000., 120000.,
       120000., 120000., 120000., 120000., 120000., 120000., 120000.,
       120000., 140000., 140000., 140000., 140000., 140000., 140000.,
       140000., 140000., 140000., 140000., 130000., 130000., 130000.,
       130000., 130000., 130000., 130000., 130000., 130000., 130000.,
       100000., 100000., 100000., 100000., 100000., 100000., 100000.,
       100000., 100000., 100000.,  90000.,  90000.,  90000.,  90000.,
        90000.,  90000.,  90000.,  90000.,  90000.,  90000.,  95000.,
        95000.,  95000.,  95000.,  95000.,  95000.,  95000.,  95000.,
        95000.,  95000.,  85000.,  85000.,  85000.,  85000.])
```

[13]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score,mean_squared_error,mean_absolute_error
lr=LinearRegression()
lr.fit(x,y)
print(lr)
```

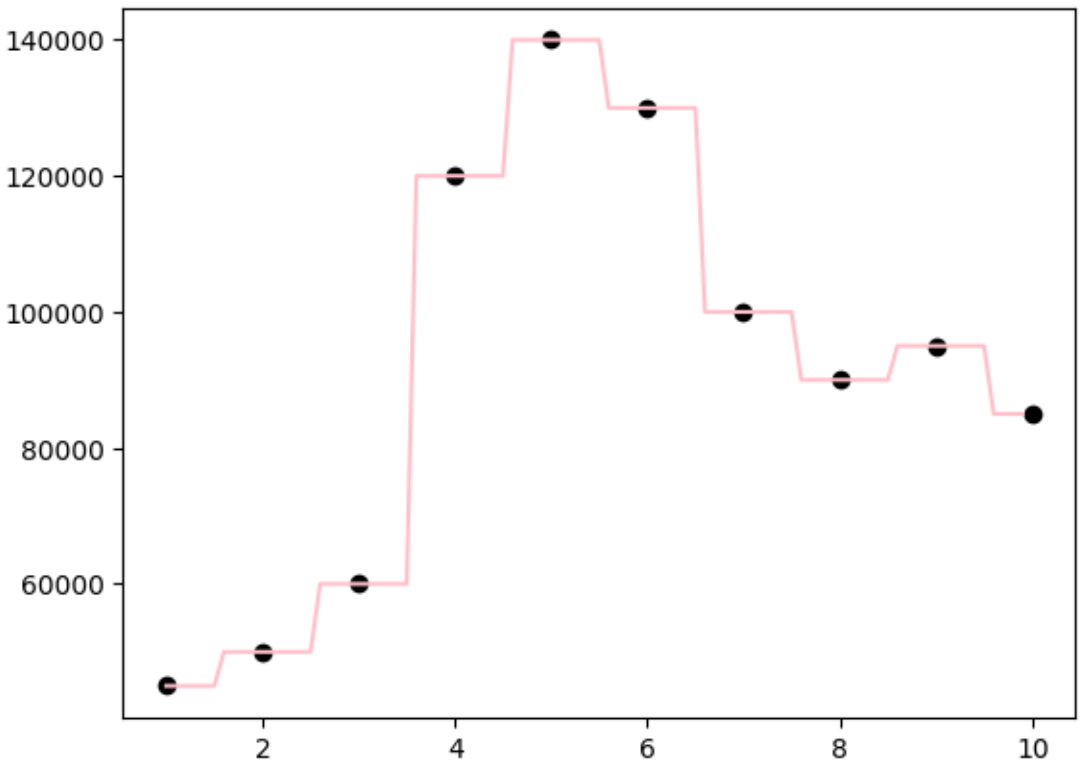
```
LinearRegression()
```

[14]:
```
lr.predict(x_grid)
```

[14]:
```
array([ 70909.09090909,  71366.66666667,  71824.24242424,  72281.81818182,
        72739.39393939,  73196.96969697,  73654.54545455,  74112.12121212,
        74569.6969697 ,  75027.27272727,  75484.84848485,  75942.42424242,
        76400.        ,  76857.57575758,  77315.15151515,  77772.72727273,
        78230.3030303 ,  78687.87878788,  79145.45454545,  79603.03030303,
        80060.60606061,  80518.18181818,  80975.75757576,  81433.33333333,
        81890.90909091,  82348.48484848,  82806.06060606,  83263.63636364,
        83721.21212121,  84178.78787879,  84636.36363636,  85093.93939394,
        85551.51515152,  86009.09090909,  86466.66666667,  86924.24242424,
        87381.81818182,  87839.39393939,  88296.96969697,  88754.54545455,
        89212.12121212,  89669.6969697 ,  90127.27272727,  90584.84848485,
        91042.42424242,  91500.        ,  91957.57575758,  92415.15151515,
        92872.72727273,  93330.3030303 ,  93787.87878788,  94245.45454545,
        94703.03030303,  95160.60606061,  95618.18181818,  96075.75757576,
        96533.33333333,  96990.90909091,  97448.48484848,  97906.06060606,
        98363.63636364,  98821.21212121,  99278.78787879,  99736.36363636,
       100193.93939394, 100651.51515152, 101109.09090909, 101566.66666667,
       102024.24242424, 102481.81818182, 102939.39393939, 103396.96969697,
       103854.54545455, 104312.12121212, 104769.6969697 , 105227.27272727,
       105684.84848485, 106142.42424242, 106600.        , 107057.57575758,
       107515.15151515, 107972.72727273, 108430.3030303 , 108887.87878788,
       109345.45454545, 109803.03030303, 110260.60606061, 110718.18181818,
       111175.75757576, 111633.33333333])
```

[15]:
```
x_grid=np.arange(min(x),max(x),0.1)
x_grid=x_grid.reshape((len(x_grid),1))
plt.scatter(x,y,color='black')
plt.plot(x_grid,dt.predict(x_grid),color='pink')
plt.show()
```

```
C:\Users\janam\AppData\Local\Temp\ipykernel_30080\3949572818.py:1:
DeprecationWarning: Conversion of an array with ndim > 0 to a scalar is
deprecated, and will error in future. Ensure you extract a single element from
your array before performing this operation. (Deprecated NumPy 1.25.)
  x_grid=np.arange(min(x),max(x),0.1)
```



[16]:
```
from sklearn.tree import DecisionTreeRegressor
split_dt=DecisionTreeRegressor(min_samples_split=8)
split_dt.fit(x,y)
print(split_dt)
```

```
DecisionTreeRegressor(min_samples_split=8)
```

[17]:
```
split_pred=split_dt.predict(x)
print(split_pred)
```

```
[ 51666.66666667  51666.66666667  51666.66666667 108571.42857143
 108571.42857143 108571.42857143 108571.42857143 108571.42857143
 108571.42857143 108571.42857143]
```

[18]:
```
r2_score(y,split_pred)
```

[18]:
```
0.7044930378263712
```

[19]:
```
arr1=np.array([3,3.5,5])
print(arr1)
```

```
[3.  3.5 5. ]
```

[20]:
```
arr1=arr1.reshape(3,1)
```

[21]:
```
arr1
```

[21]:
```
array([[3. ],
       [3.5],
       [5. ]])
```

[22]:
```
y_pred1=dt.predict(arr1)
print(y_pred1)
```

```
[ 60000.  60000. 140000.]
```

[23]:
```
y_pred2=split_dt.predict(arr1)
print(y_pred2)
```

```
[ 51666.66666667  51666.66666667 108571.42857143]
```

[24]:
```
import pandas as pd
```

[25]:
```
df=pd.read_csv('diabetes_dataset.csv')
df.head()
```

[25]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	\
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	

	DiabetesPedigreeFunction	Age	Outcome
0	0.627	50	1
1	0.351	31	0
2	0.672	32	1
3	0.167	21	0
4	2.288	33	1

[26]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
```

```
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

[27]:
```
x=df.drop(['Outcome'],axis=1)
x
```

[27]:
```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6
1              1       85             66             29        0  26.6
2              8      183             64              0        0  23.3
3              1       89             66             23       94  28.1
4              0      137             40             35      168  43.1
..           ...      ...            ...            ...      ...  ...
763           10      101             76             48      180  32.9
764            2      122             70             27        0  36.8
765            5      121             72             23      112  26.2
766            1      126             60              0        0  30.1
767            1       93             70             31        0  30.4

     DiabetesPedigreeFunction  Age
0                       0.627   50
1                       0.351   31
2                       0.672   32
3                       0.167   21
4                       2.288   33
..                        ...  ...
763                     0.171   63
764                     0.340   27
765                     0.245   30
766                     0.349   47
767                     0.315   23

[768 rows x 8 columns]
```

[28]:
```
y=df.Outcome
y
```

[28]:
```
0      1
1      0
2      1
3      0
4      1
```

```
      ..
763    0
764    0
765    0
766    1
767    0
Name: Outcome, Length: 768, dtype: int64
```

[29]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=1)
```

[30]:
```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(614, 8)
(154, 8)
(614,)
(154,)
```

[31]:
```
model=DecisionTreeClassifier()
model=model.fit(x_train,y_train)
y_pred=model.predict(x_test)
```

[32]:
```
y_pred
```

[32]:
```
array([0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0,
       0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1,
       0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0,
       0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0,
       1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0],
      dtype=int64)
```

[33]:
```
from sklearn import metrics
print("Accuracy:",metrics.accuracy_score(y_test,y_pred)*100)
```

```
Accuracy: 70.12987012987013
```

[34]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

[34]:
```
array([[79, 20],
       [26, 29]], dtype=int64)
```

[35]:
```
print("Accuracy:",((82+27)/154))
```

```
Accuracy: 0.7077922077922078
```

[36]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.75      0.80      0.77        99
           1       0.59      0.53      0.56        55

    accuracy                           0.70       154
   macro avg       0.67      0.66      0.67       154
weighted avg       0.70      0.70      0.70       154
```

[37]:
```
model.predict([[6,2,4,6,8,4,45,76]])
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py:439: UserWarning: X
does not have valid feature names, but DecisionTreeClassifier was fitted with
feature names
  warnings.warn(
```

[37]:
```
array([0], dtype=int64)
```

[38]:
```
pip install pydotplus
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pydotplus in
c:\users\janam\appdata\roaming\python\python311\site-packages (2.0.2)
Requirement already satisfied: pyparsing>=2.0.1 in
c:\users\janam\appdata\roaming\python\python311\site-packages (from pydotplus)
(3.1.2)
Note: you may need to restart the kernel to use updated packages.
```

[39]:
```
from sklearn.tree import export_graphviz
import six
import sys
sys.modules['sklearn.externals.six']=six
from sklearn.externals.six import StringIO
from IPython.display import Image
import pydotplus
```

[40]:
```
features=x.columns
features
```

[40]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age'],
      dtype='object')
```

[42]:
```
!pip install graphviz
```

```
Defaulting to user installation because normal site-packages is not writeable
Collecting graphviz
  Downloading graphviz-0.20.3-py3-none-any.whl.metadata (12 kB)
Downloading graphviz-0.20.3-py3-none-any.whl (47 kB)
   ---------------------------------------- 0.0/47.1 kB ? eta -:--:--
   -------- ------------------------------- 10.2/47.1 kB ? eta -:--:--
   ---------------- ----------------------- 20.5/47.1 kB 320.0 kB/s eta 0:00:01
   ------------------------- -------------- 30.7/47.1 kB 325.1 kB/s eta 0:00:01
   ---------------------------------------- 47.1/47.1 kB 295.1 kB/s eta 0:00:00
Installing collected packages: graphviz
Successfully installed graphviz-0.20.3
```

[6]:
```
from sklearn.externals.six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus
dot_data = StringIO()
export_graphviz(model, out_file=dot_data,filled=True,␣
 ↪rounded=True,special_characters=True, feature_names =␣
 ↪features,class_names=['0','1'])
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('diabetes_dataset.csv')
Image(graph.create_png())
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Cell In[6], line 1
----> 1 from sklearn.externals.six import StringIO
      2 from IPython.display import Image
      3 from sklearn.tree import export_graphviz

ModuleNotFoundError: No module named 'sklearn.externals.six'
```

[7]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

[10]:
```
df=pd.read_csv('Position Salary .csv')
print(df)
```

```
            Position  Level  Salary
0   Busienss Analyst      1   45000
1  Junior Consultant      2   50000
2  Senior Consultant      3   60000
3            Manager      4  120000
4    Country Manager      5  140000
5     Region Manager      6  130000
6            Partnor      7  100000
7     Senior Partnor      8   90000
8            C Level      9   95000
9               CEO      10  85000
```

[11]:
```
x=df.iloc[:,1:2].values
print(x)
```

```
[[ 1]
 [ 2]
 [ 3]
 [ 4]
 [ 5]
 [ 6]
 [ 7]
 [ 8]
 [ 9]
 [10]]
```

[12]:
```
y=df.iloc[:,2].values
print(y)
```

```
[ 45000  50000  60000 120000 140000 130000 100000  90000  95000  85000]
```

[13]:
```
from sklearn.ensemble import RandomForestRegressor
rf=RandomForestRegressor()
rf.fit(x,y)
```

[13]:
```
RandomForestRegressor()
```

[15]:
```
y_pred=rf.predict(x)
print(y_pred)
```

```
[ 48000.  49950.  62550. 106400. 131000. 129600. 106250.  94400.  93000.
  88500.]
```

[17]:
```
from sklearn.metrics import r2_score
r2_score=r2_score(y,y_pred)
r2_score
```

[17]:
```
0.9630875420875421
```

[1]:
```
pip install opencv-python
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: opencv-python in
c:\users\janam\appdata\roaming\python\python311\site-packages (4.9.0.80)
Requirement already satisfied: numpy>=1.21.2 in
c:\users\janam\appdata\roaming\python\python311\site-packages (from opencv-
python) (1.26.4)
Note: you may need to restart the kernel to use updated packages.
```

[2]:
```
import cv2
```

[3]:
```
img=cv2.imread("animals-7696695_640.jpg")
cv2.imshow("image:",img)
cv2.waitKey(0)
```

[3]:
```
-1
```

[4]:
```
half=cv2.resize(img,(0,0),fx=0.5,fy=0.5)
cv2.imshow("Resized image:",half)
cv2.waitKey(0)
```

[4]:
```
-1
```

[5]:
```
bigger=cv2.resize(img,(1000,1000))
cv2.imshow("Resized image:",bigger)
cv2.waitKey(0)
```

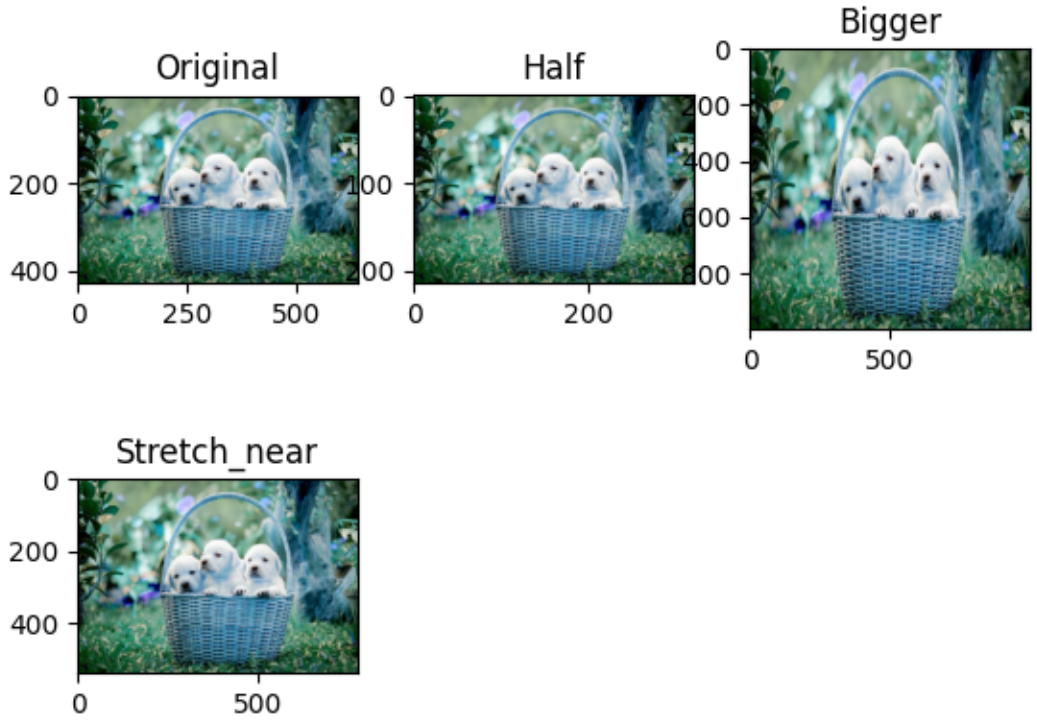
[5]:
```
-1
```

[6]:
```
stretch_near=cv2.resize(img,(780,540),interpolation=cv2.INTER_NEAREST)
cv2.imshow("Resized image:",stretch_near)
cv2.waitKey(0)
```

[6]:
```
-1
```

[7]:
```
import matplotlib.pyplot as plt
Titles=["Original","Half","Bigger","Stretch_near"]
images=[img,half,bigger,stretch_near]
for i in range(4):
    plt.subplot(2,3,i+1)
```

```
    plt.title(Titles[i])
    plt.imshow(images[i])
```



[9]:
```
import cv2
import numpy as np
image=cv2.imread("animals-7696695_640.jpg")
cv2.imshow("image:",image)
cv2.waitKey(0)
Gaussian=cv2.GaussianBlur(image,(7,7),0)
cv2.imshow('Gaussian Blurring:',Gaussian)
cv2.waitKey(0)
median=cv2.medianBlur(image,5)
cv2.imshow("Median Blurring:",median)
cv2.waitKey(0)
bilateral=cv2.bilateralFilter(image,5,75,75)
cv2.imshow("Bilateral :",bilateral)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

[11]:
```
image=cv2.imread("animals-7696695_640.jpg")
edge=cv2.Canny(image,100,150)
cv2.imshow("Edge image:",edge)
cv2.waitKey(0)
```

[11]: -1

[13]:
```
(rows,cols)=img.shape[:2]
print(rows,cols)
M=cv2.getRotationMatrix2D((100,100),90,1)
res=cv2.warpAffine(img,M,(200,200))
cv2.imshow("Res:",res)
cv2.waitKey(0)
```

```
427 640
```

[13]: -1

[18]:
```
import cv2
alg='.xml'
cascade=cv2.CascaderClassifier(alg)
cam=cv2.VideoCapture(0)
while True:
    _,img=cam.read()
    grayimg=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    face=cascade.detectMultiScale(grayImg)
    for(x,y,w,h) in face:
        cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),2)
    cv2.imshow("FaceDetect",img)
    key=cv2.waitKey(1)
    if key==81 or key==113:
        break
        cv2.destroyAllWindows()
        cam.release()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[18], line 3
      1 import cv2
      2 alg='haarcascade_frontalface_default.xml'
----> 3 cascade=cv2.CascaderClassifier(alg)
      4 cam=cv2.VideoCapture(0)
      5 while True:

AttributeError: module 'cv2' has no attribute 'CascaderClassifier'
```

[16]:
```
from cv2 import *
cam_port=0
cam=cv2.VideoCapture(cam_port)
result,image=cam.read()
if result:
```

```
    cv2.imshow("pst",image)
   # cv2.imwrite("pst",image)
    cv2.waitKey(0)
    cv2.destroyWindow("pst")
else:
    print("No image")
```

```
---------------------------------------------------------------------------
error                                     Traceback (most recent call last)
Cell In[16], line 9
      7    # cv2.imwrite("pst",image)
      8     cv2.waitKey(0)
----> 9     cv2.destroyWindow("pst")
     10 else:
     11     print("No image")

error: OpenCV(4.9.0) D:
 ↪\a\opencv-python\opencv-python\opencv\modules\highgui\src\window_w32.cpp:1261 ␣
 ↪error: (-27:Null pointer) NULL window: 'pst' in function 'cvDestroyWindow'
```

[ ]: