cat_v_dog_KNN

September 16, 2023

[1]:
```
import cv2 as cv
import numpy as np
from matplotlib import pyplot as plt
import os
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import GridSearchCV
import pandas as pd
```

[2]:
```
def load_images_from_folder(folder):
    images = []
    for file in os.listdir(folder):
        img = cv.imread(os.path.join(folder,file))
        if(img is not None):
            images.append(img)
    return images
```

[3]:
```
def resize_and_flatten(img, target_size=(64,64)):
    resized_img = cv.resize(img, target_size)
    flattened_img = resized_img.reshape(target_size[0] * target_size[1] * 3)
    return flattened_img
```

[4]:
```
cat_data = []
for img in load_images_from_folder("../data/cat"):
    cat_data.append(resize_and_flatten(img))
dog_data = []
for img in load_images_from_folder("../data/dog"):
    dog_data.append(resize_and_flatten(img))
```

[5]:
```
X = []
y = []

for entity in cat_data:
    X.append(entity)
    y.append("cat")

for entity in dog_data:
    X.append(entity)
```

```
    y.append("dog")


np.array(X).shape, np.array(y).shape
```

[5]:
```
((160, 12288), (160,))
```

[6]:
```
kNN_model = KNeighborsClassifier()
```

[7]:
```
kNN_model.get_params()
```

[7]:
```
{'algorithm': 'auto',
 'leaf_size': 30,
 'metric': 'minkowski',
 'metric_params': None,
 'n_jobs': None,
 'n_neighbors': 5,
 'p': 2,
 'weights': 'uniform'}
```

[8]:
```
model = GridSearchCV(estimator=kNN_model, param_grid={'n_neighbors':
 ↪[1,3,5,7,8,11]},cv=3)
model.fit(X,y)
pd.DataFrame(model.cv_results_)
```

[8]:

	mean_fit_time	std_fit_time	mean_score_time	std_score_time	\
0	0.000903	0.000416	0.014124	0.000526	
1	0.000429	0.000018	0.014610	0.000335	
2	0.000413	0.000004	0.012484	0.000854	
3	0.000417	0.000010	0.013910	0.000614	
4	0.000412	0.000001	0.012041	0.000625	
5	0.000408	0.000006	0.012877	0.001024	

	param_n_neighbors	params	split0_test_score	\
0	1	{'n_neighbors': 1}	0.481481	
1	3	{'n_neighbors': 3}	0.500000	
2	5	{'n_neighbors': 5}	0.611111	
3	7	{'n_neighbors': 7}	0.518519	
4	8	{'n_neighbors': 8}	0.500000	
5	11	{'n_neighbors': 11}	0.462963	

	split1_test_score	split2_test_score	mean_test_score	std_test_score	\
0	0.509434	0.509434	0.500116	0.013177	
1	0.490566	0.584906	0.525157	0.042424	
2	0.528302	0.716981	0.618798	0.077220	
3	0.528302	0.660377	0.569066	0.064690	
4	0.509434	0.547170	0.518868	0.020380	

```
5           0.509434           0.566038         0.512812        0.042148
```

```
   rank_test_score
0                6
1                3
2                1
3                2
4                4
5                5
```