```
Start coding or generate with AI.
```

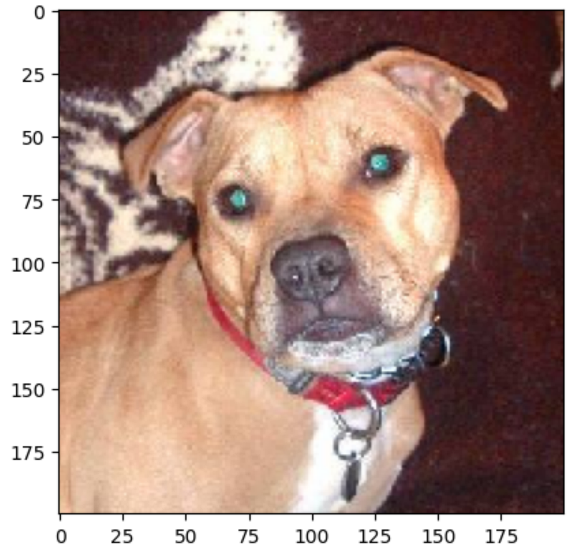
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image # If you need other image preprocessing functions from keras
```

```
img = image.load_img("/2.jpg",target_size=(200,200))
```

```
import matplotlib.pyplot as plt
```

```
plt.imshow(img)
```

```
<matplotlib.image.AxesImage at 0x78258eaa7890>
```



```
type(img)
```

```
PIL.Image.Image
def __init__() -> None

/usr/local/lib/python3.11/dist-packages/PIL/Image.py
This class represents an image object.  To create
:py:class:`~PIL.Image.Image` objects, use the appropriate factory
functions.  There's hardly ever any reason to call the Image constructor
directly.
```

```
datagenerate = ImageDataGenerator(
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,

)
```

```
img = image.img_to_array(img)
```

Double-click (or enter) to edit

```
type(img)
```

```
numpy.ndarray
```

```
img.shape
```

```
(200, 200, 3)
```

```
input_batch = img.reshape(1,200,200,3)
```

Start coding or generate with AI.

Start coding or generate with AI.

```
import os

# ... (rest of your code) ...

# Before calling datagenerate.flow(), create the directory if it doesn't exist
if not os.path.exists('aug'):
    os.makedirs('aug')

i = 0
for output in datagenerate.flow(input_batch, batch_size=1, save_to_dir='aug'):
    i = i + 1
    if i == 10:
        break # Changed 'brake' to 'break'
```

```
input_batch.shape
```

```
(1, 200, 200, 3)
```

Start coding or generate with AI.