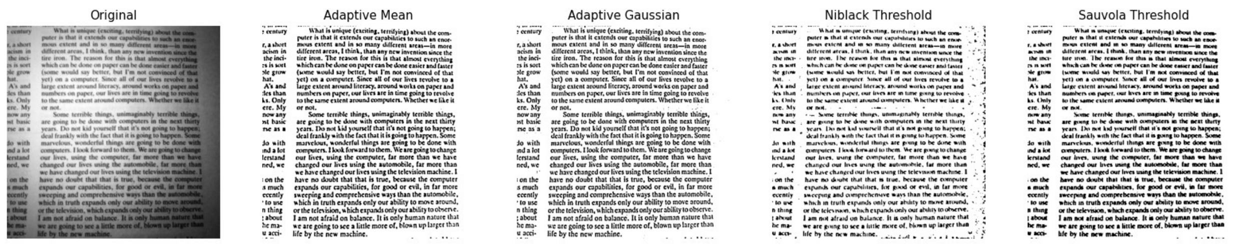
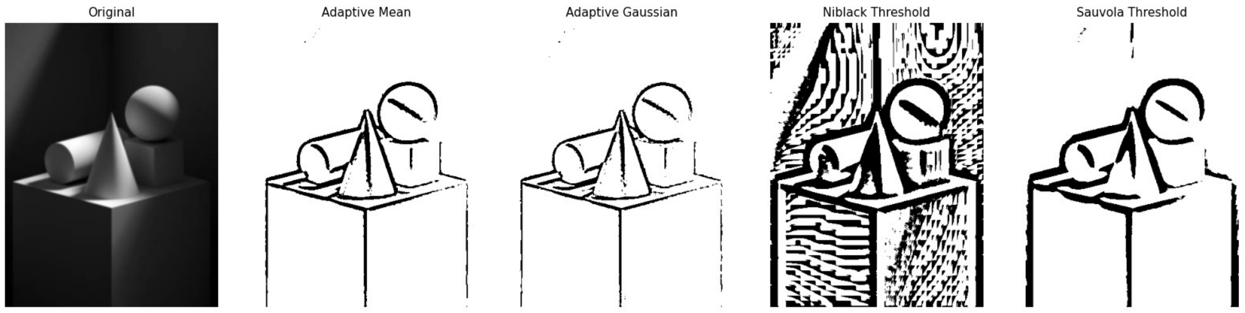
```
import matplotlib
import matplotlib.pyplot as plt
import cv2 as cv
import os

from skimage.filters import threshold_niblack, threshold_sauvola

matplotlib.rcParams['font.size'] = 9

# Define image paths
image_path = "./shadowed_text_2.jpg"

# Loop over each image path

img = cv.imread(image_path, cv.IMREAD_GRAYSCALE)
assert os.path.exists(image_path), f"File {image_path} could not be
read, check with os.path.exists()"

# Apply Gaussian filtering
image = cv.GaussianBlur(img, (5, 5), 0)

# Apply adaptive thresholding methods
adaptive_thresh_mean = cv.adaptiveThreshold(image, 255,
cv.ADAPTIVE_THRESH_MEAN_C, cv.THRESH_BINARY, 11, 10)
adaptive_thresh_gaussian = cv.adaptiveThreshold(image, 255,
cv.ADAPTIVE_THRESH_GAUSSIAN_C, cv.THRESH_BINARY, 11, 5)

# Apply Niblack and Sauvola methods
window_size = 25
thresh_niblack = threshold_niblack(image, window_size=window_size,
k=1)
thresh_sauvola = threshold_sauvola(image, window_size=window_size,
k=0.2)

binary_niblack = image > thresh_niblack
binary_sauvola = image > thresh_sauvola

# Display the results
plt.figure(figsize=(20, 5))

plt.subplot(1, 5, 1)
plt.imshow(image, cmap=plt.cm.gray)
plt.title('Original')
plt.axis('off')

plt.subplot(1, 5, 2)
plt.imshow(adaptive_thresh_mean, cmap=plt.cm.gray)
plt.title('Adaptive Mean')
plt.axis('off')
plt.subplot(1, 5, 3)
plt.imshow(adaptive_thresh_gaussian, cmap=plt.cm.gray)
plt.title('Adaptive Gaussian')
plt.axis('off')

plt.subplot(1, 5, 4)
plt.imshow(binary_niblack, cmap=plt.cm.gray)
plt.title('Niblack Threshold')
plt.axis('off')

plt.subplot(1, 5, 5)
plt.imshow(binary_sauvola, cmap=plt.cm.gray)
plt.title('Sauvola Threshold')
plt.axis('off')

plt.show()
```



```
import matplotlib
import matplotlib.pyplot as plt
import cv2 as cv
import os

from skimage.filters import threshold_niblack, threshold_sauvola

matplotlib.rcParams['font.size'] = 9

# Define image paths
image_path = "./shadowed_object_1.jpg"

# Loop over each image path

img = cv.imread(image_path, cv.IMREAD_GRAYSCALE)
assert os.path.exists(image_path), f"File {image_path} could not be
read, check with os.path.exists()"

# Apply Gaussian filtering
image = cv.GaussianBlur(img, (5, 5), 0)

# Apply adaptive thresholding methods
adaptive_thresh_mean = cv.adaptiveThreshold(image, 255,
cv.ADAPTIVE_THRESH_MEAN_C, cv.THRESH_BINARY, 11, 3)
adaptive_thresh_gaussian = cv.adaptiveThreshold(image, 255,
cv.ADAPTIVE_THRESH_GAUSSIAN_C, cv.THRESH_BINARY, 11, 2)

# Apply Niblack and Sauvola methods
window_size = 25
thresh_niblack = threshold_niblack(image, window_size=window_size,
k=0.2)
thresh_sauvola = threshold_sauvola(image, window_size=window_size,
k=0.2)

binary_niblack = image > thresh_niblack
binary_sauvola = image > thresh_sauvola

# Display the results
plt.figure(figsize=(20, 5))

plt.subplot(1, 5, 1)
plt.imshow(image, cmap=plt.cm.gray)
plt.title('Original')
plt.axis('off')

plt.subplot(1, 5, 2)
plt.imshow(adaptive_thresh_mean, cmap=plt.cm.gray)
plt.title('Adaptive Mean')
plt.axis('off')

plt.subplot(1, 5, 3)
plt.imshow(adaptive_thresh_gaussian, cmap=plt.cm.gray)
plt.title('Adaptive Gaussian')
plt.axis('off')

plt.subplot(1, 5, 4)
plt.imshow(binary_niblack, cmap=plt.cm.gray)
plt.title('Niblack Threshold')
plt.axis('off')

plt.subplot(1, 5, 5)
plt.imshow(binary_sauvola, cmap=plt.cm.gray)
plt.title('Sauvola Threshold')
plt.axis('off')

plt.show()
```


Original
Adaptive Mean
Adaptive Gaussian
Niblack Threshold
Sauvola Threshold