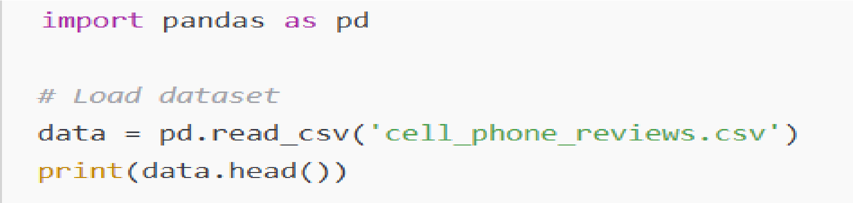
# Data Collection and Preprocessing Phase

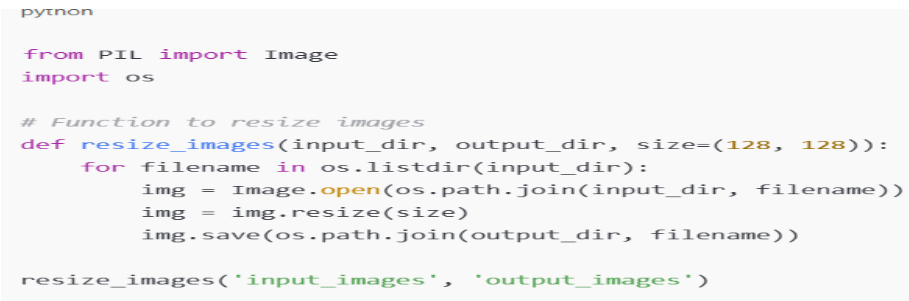
| | |
|---|---|
| Date | 15 October 2024 |
| Team ID | SWTID1726832093 |
| Project Title | Analysis of Amazon Cell Phone Reviews Using NLP Technique |
| Maximum Marks | 6 Marks |

**Preprocessing Template**

The images will be preprocessed by resizing, normalizing, augmenting, denoising, adjusting contrast, detecting edges, converting color space, cropping, batch normalizing, and whitening data. These steps will enhance data quality, promote model generalization, and improve convergence during neural network training, ensuring robust and efficient performance across various computer vision tasks.

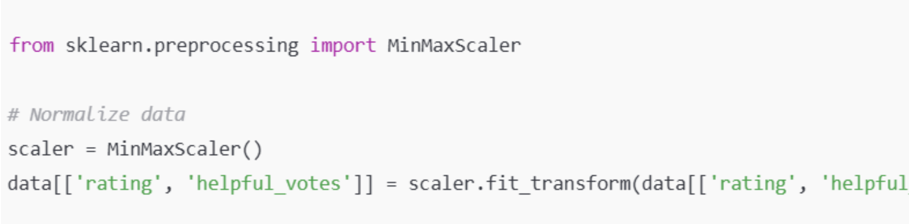
| **Section** | **Description** |
|---|---|
| Data Overview | the dataset consists of images of cell phones along with corresponding reviews from amazon. the reviews are in text format and will be analyzed using nlp techniques to extract insights regarding customer sentiment, frequently mentioned features, and overall trends in consumer preferences. |
| Resizing | Images will be resized to a target size (e.g., 224x224 pixels) to maintain consistency for model input. |
| Normalization | Pixel values will be normalized to a specific range (e.g., 0 to 1) to facilitate faster convergence during model training. |
| Data Augmentation | Apply augmentation techniques such as flipping, rotation, shifting, zoo or shearing. |

<table>
<tr><td>Denoising</td><td>Denoising filters will be applied to reduce noise<br>in images, enhancing the quality of the data.</td></tr>
<tr><td>==============================<br>==============================</td><td>==========================================================</td></tr>
<tr><td></td><td></td></tr>
<tr><td></td><td></td></tr>
<tr><td></td><td></td></tr>
<tr><td colspan="2"><b>Data Preprocessing Code Screenshots</b></td></tr>
<tr><td>Loading Data</td><td>
<pre><code>import pandas as pd

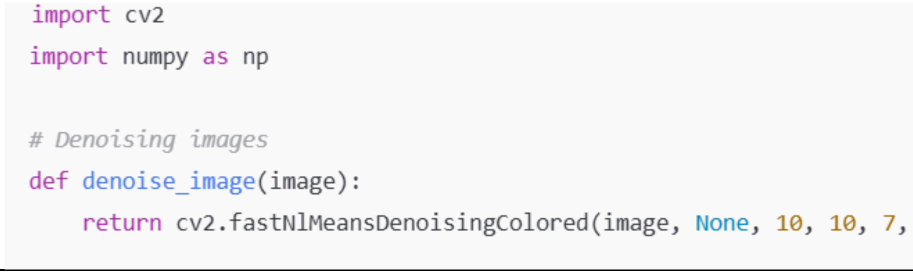
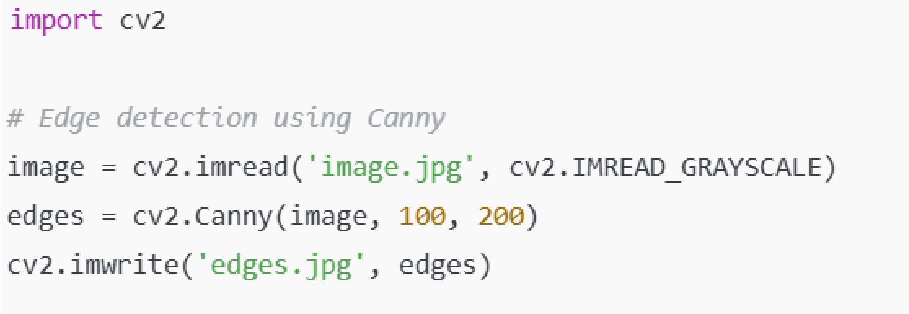
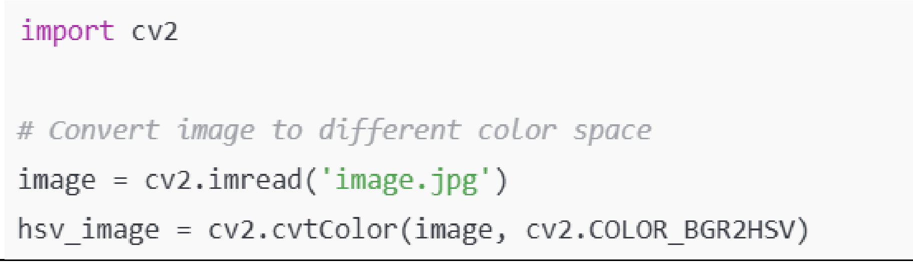
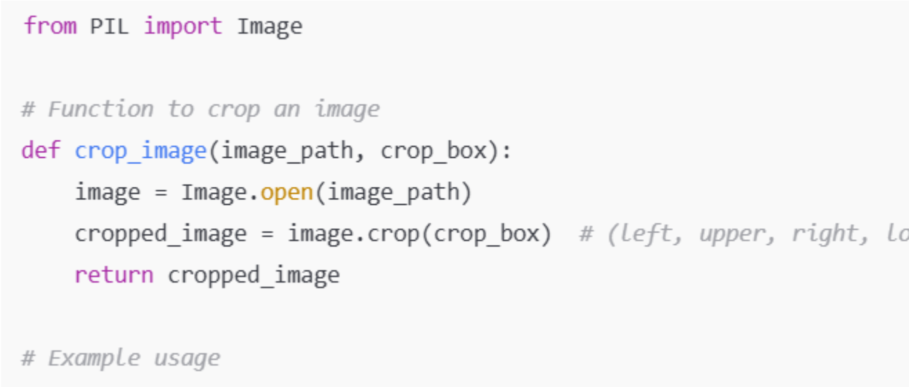
# Load dataset
data = pd.read_csv('cell_phone_reviews.csv')
print(data.head())</code></pre></td></tr>
<tr><td>Resizing</td><td>
<pre><code>python

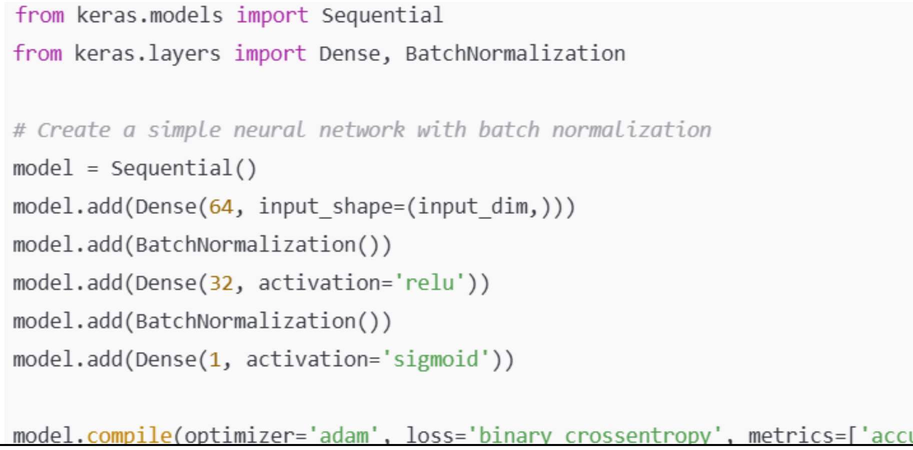
from PIL import Image
import os

# Function to resize images
def resize_images(input_dir, output_dir, size=(128, 128)):
    for filename in os.listdir(input_dir):
        img = Image.open(os.path.join(input_dir, filename))
        img = img.resize(size)
        img.save(os.path.join(output_dir, filename))

resize_images('input_images', 'output_images')</code></pre></td></tr>
<tr><td>Normalization</td><td>
<pre><code>from sklearn.preprocessing import MinMaxScaler

# Normalize data
scaler = MinMaxScaler()
data[['rating', 'helpful_votes']] = scaler.fit_transform(data[['rating', 'helpful</code></pre></td></tr>
</table>

| | |
|---|---|
| Denoising | <br>import cv2<br>import numpy as np<br><br># Denoising images<br>def denoise_image(image):<br>    return cv2.fastNlMeansDenoisingColored(image, None, 10, 10, 7, |
| Edge Detection | <br>import cv2<br><br># Edge detection using Canny<br>image = cv2.imread('image.jpg', cv2.IMREAD_GRAYSCALE)<br>edges = cv2.Canny(image, 100, 200)<br>cv2.imwrite('edges.jpg', edges) |
| Color Space Conversion | <br>import cv2<br><br># Convert image to different color space<br>image = cv2.imread('image.jpg')<br>hsv_image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV) |
| Image Cropping | <br>from PIL import Image<br><br># Function to crop an image<br>def crop_image(image_path, crop_box):<br>    image = Image.open(image_path)<br>    cropped_image = image.crop(crop_box)  # (left, upper, right, lo<br>    return cropped_image<br><br># Example usage |

<table>
<tr>
<td>Batch Normalization</td>
<td>

<pre>
from keras.models import Sequential
from keras.layers import Dense, BatchNormalization

# Create a simple neural network with batch normalization
model = Sequential()
model.add(Dense(64, input_shape=(input_dim,)))
model.add(BatchNormalization())
model.add(Dense(32, activation='relu'))
model.add(BatchNormalization())
model.add(Dense(1, activation='sigmoid'))

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accu
</pre>
</td>
</tr>
</table>