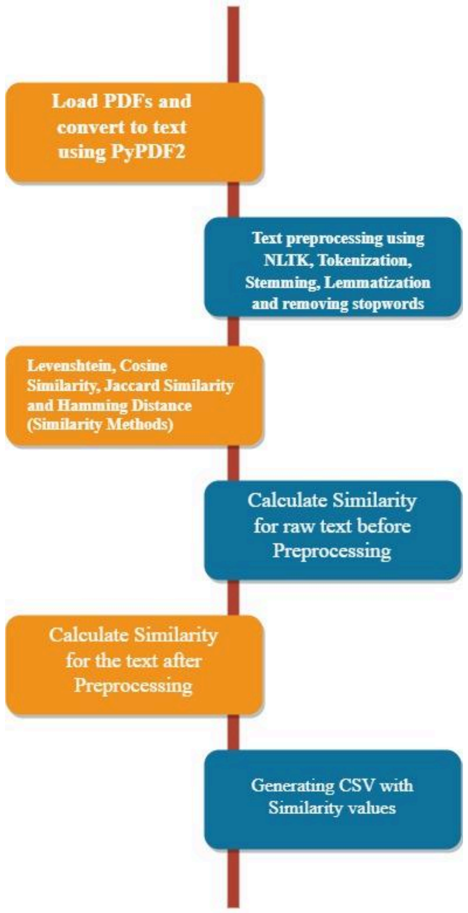
Similarity of PDF having both Text and Images.

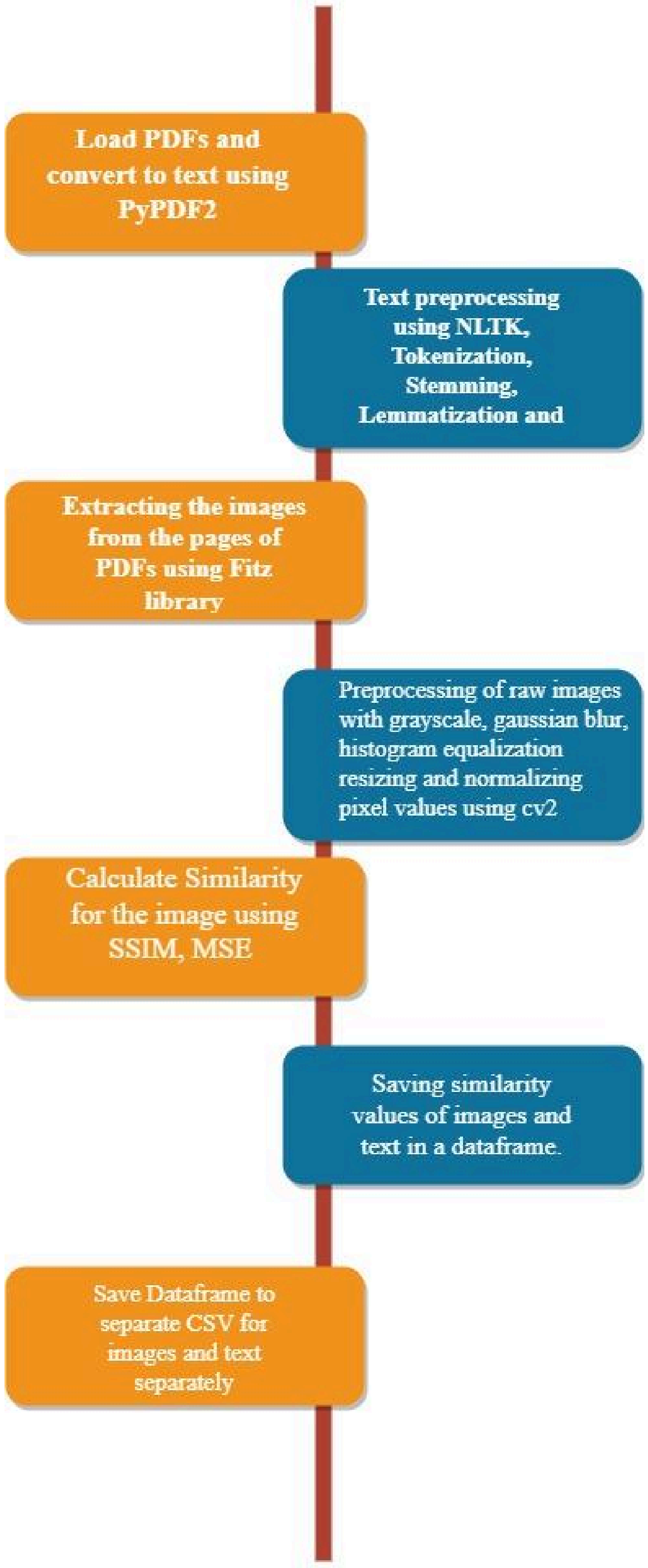
Similarity of PDF with text.(text_similarity.py)

1. First, we have taken pdfs containing ‘Terms & Uses’ from ECommerce websites like Sephora,Jcrew,Gymshark,etc., and stored them in a folder.
2. For reading each file, executed a python script to load them using PyPDF2 library and stored them in a dataframe in lower case.
3. Next, we have implemented a series of different techniques to get an idea of how the values of similarity indexes are looking like for any 2 files using various methods like Levenshtein distance, Cosine similarity, Jaccard similarity, and Hamming distance from Levenshtein, textdistance, sklearn.
4. The preprocessing on the previously obtained dataframe which contained lowercase letters has been done with removing non-alphanumeric characters, tokenization, eliminating stopwords, Lemmatization and stemming from PorterStemmer, NLTK libraries, which then stored as a pickle file.
5. Now, a csv file has been generated with columns having the names of the 2 selected pdfs, similarity value of a method without the preprocessing and similarity value of a method with preprocessing.


Load PDFs and convert to text using PyPDF2
Text preprocessing using NLTK, Tokenization, Stemming, Lemmatization and removing stopwords
Levenshtein, Cosine Similarity, Jaccard Similarity and Hamming Distance (Similarity Methods)
Calculate Similarity for raw text before Preprocessing
Calculate Similarity for the text after Preprocessing
Generating CSV with Similarity values

Similarity of PDF with images between text.(image_text_similarity.py)

1. Here with the help of PyPDF2 library we loaded the pdfs and extracted the text then made it into lowercase and saved it in a dataframe and
2. We also extracted the image from a page of the pdf which then stored in a folder (ex. images) by importing the fitz library.
3. So after the images have been extracted, we have done preprocessing of the image using grayscale, gaussian blur, histogram equalization resizing and normalizing pixel values by importing from cv2 library.
4. Then we calculated the image similarity between 2 images using SSIM and MSE from skimage.matrices library.
5. Simultaneously, we calculated the similarity value between the text using the same techniques like the above and we are saving it into a csv file.
6. After going through the whole process for the image, we are saving the images and the similarity value into the dataframe and finally saving it into a different csv file naming ends with '_image'


Load PDFs and convert to text using PyPDF2
Text preprocessing using NLTK, Tokenization, Stemming, Lemmatization and
Extracting the images from the pages of PDFs using Fitz library
Preprocessing of raw images with grayscale, gaussian blur, histogram equalization resizing and normalizing pixel values using cv2
Calculate Similarity for the image using SSIM, MSE
Saving similarity values of images and text in a dataframe.
Save Dataframe to separate CSV for images and text separately

Streamlit integration to display similar images of the selected image.

1. Here, we directly called a function from the image similarity checking function from the image_text_similarity.py so as to obtain the images to display in the streamlit.
2. We can select the name of a photo from the set of images from extract_images such that the selected image will be displayed on the left side and similar images along with their similarity values are displayed on the right side using st.columns

Streamlit integration to display the similarity of text in the different pdfs.

1. Here, we directly called a function from the text similarity checking function from the text_similarity.py so as to obtain the similarity values of different pdfs to display in the streamlit.
2. In this we mainly added a filter to select the type of similarity checking for the raw data and preprocessed data.