DSC160 Data Science and the Arts - Twomey - Spring 2020 - dsc160.roberttwomey.com (http://dsc160.roberttwomey.com)

Exercise 1: Reading Image Archives (Web-Scraping and Basic Features)

This exercise takes you through the a coarse image-feature based analysis of a famous Abstract Expressionist painter, Mark Rothko (https://www.biography.com/artist/mark-rothko). Technically, you will build a full workflow from image retrieval from an online archive -> calculation of image features -> visualization of results. Finally, it asks you to reproduce a similar result using a small image data set of your choice.

The exercise is broken down into two parts:

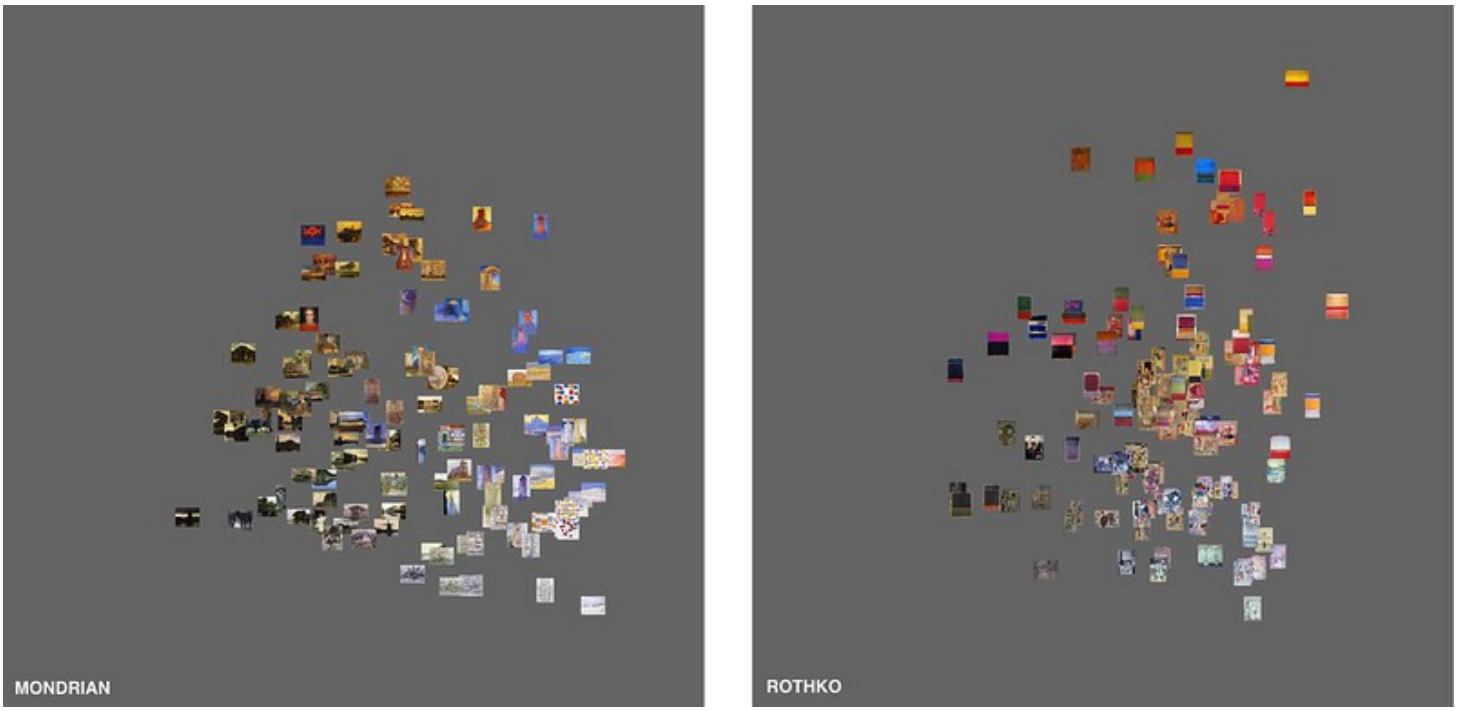
- Part 1. A replication of an analysis by the Software Studies Initiative/Lev Manovich of Mark Rothko's paintings.
- Part 2. The second section asks you to extend this work, applying the same methods to analyze an image set (n <= 100) of your choosing.

Once you have completed both parts, you will submit your completed notebook as a pdf to gradescope for grading.

Part 1: Plotting Rothko

(30 pts total)

Mark Rothko (https://www.biography.com/artist/mark-rothko) is a celebrated Abstract Expressionist painter known for his large color field abstractions. Some historians describe a progression towards darker, less colorful compositions over the course of his life. Here, we will recreating plots similar to the plots below from the Software Studies Initiative, showing a distribution of color and brightness within his body of work.



Data: 128 paintings by Piet Mondrian (1905-1917); 123 paintings by Mark Rothko (1938-1953). Mapping: The two image plots are placed side by side. In each plot: X-axis: brightness mean; Y-axis: saturation mean.

From Mondrian vs Rothko: footprints and evolution in style space (http://lab.softwarestudies.com/2011/06/mondrian-vs-rothko-footprints-and.html)

1A. Retrieving Data from a Visual Archive

(5 points)

First you need to retreive images of Rothko's paintings from an online cultural archive. WikiArt has 163 of Rothko's paintings: https://www.wikiart.org/en/mark-rothko (https://www.wikiart.org/en/mark-rothko). We will retrieve all of these images and store them locally.

You can model your code on our example notebook for scraping images from WikiArt: ../examples/scrape-wikiart.ipynb (../examples/scrape-wikiart.ipynb)

In [1]:
```
# import libraries
from bs4 import BeautifulSoup
import os
import requests

#set up data paths
DATA_DIR = '../data/'
ARTIST_URL = 'https://www.wikiart.org/en/{artist}/all-works/text-list'
PAINTING_URL = 'https://www.wikiart.org{painting_path}'

#make folder for paintings we want to download
if not os.path.exists(DATA_DIR):
    os.makedirs(DATA_DIR)

#get list of paintings
artist_name = 'mark-rothko'
url_query = ARTIST_URL.format(artist=artist_name)
artist_page = requests.get(url_query)

# check for request error
try:
    artist_page.raise_for_status()
except requests.exceptions.HTTPError as e:
    print("Error trying to retrieve {}".format(artist_page.url))
    raise e

#call web scrapper
soup = BeautifulSoup(artist_page.text, 'lxml')

#create image storage directory for artist if it doesn't exist
IMAGE_DIR = os.path.join(DATA_DIR, artist_name)
if not os.path.exists(IMAGE_DIR):
    os.makedirs(IMAGE_DIR)

#make array for paths
painting_paths = []

# retreive all rows in painting-list
for li in soup.find_all('li', {'class': 'painting-list-text-row'}):

    # retrieve all links in the current row
    for link in li.find_all('a'):
        href = link.get('href')
        # store in dictionary
        painting_paths.append(href)

print(len(painting_paths))
# painting_paths
```

```
#store number of paintings
num_paintings = len(painting_paths);

#download
def download_and_save(painting_url):
    r_painting_page = requests.get(painting_url)
    soup = BeautifulSoup(r_painting_page.text, 'lxml')
    for img in soup.find_all('img', {'class': 'ms-zoom-cursor'}):
        img_url = img['src']
        img_url = img_url.split('!')[0]
        filename = img_url.split('/')[-1]

        outfile = os.path.join(IMAGE_DIR, filename)
        if not os.path.exists(outfile):
            print("downloading {}: {}".format(filename, img_url))
            r = requests.get(img_url, outfile)
            with open(outfile, 'wb') as f:
                f.write(r.content)
        else:
            #print("skipping {}".format(filename))
            pass

#use our custom function to download paintings
for path in painting_paths:
    painting_path = PAINTING_URL.format(painting_path=path)
    download_and_save(painting_path)
```

163

1B. Calculating Basic Image Features

(10 points)

This section presumes you have already scraped/downloaded your set of images (n of approx. 160). In this section you will iterate over your downloaded images and calculate a number of image statistics, saving the results in a pandas dataframe.

First, write a function `calc_stats()` that takes filename as an input and returns a list of image stats, including:

- image width (pixels)
- image height (pixels)
- mean hue
- mean saturation
- mean value (brightness)

(for examples of how to calculate basic image statistics, see ../examples/basic-image-stats.ipynb (../examples/basic-image-stats.ipynb))

In [2]:
```
%matplotlib inline

import numpy as np
from skimage import io
import skimage
#import os
#from skimage import data
from skimage.color import rgb2hsv

#img_stats = np.zeros((5, 1))
#print(img_stats)

def calc_stats(filename):
  # function calc_stats returns width, height, hue, saturation and mea
n value (brightness) in an array

    #vector for our output
    img_stats = np.zeros((5, 1))

    my_image = io.imread(filename) #read the image
    img_stats[0] = my_image.shape[0] #get the width
    img_stats[1] = my_image.shape[1] #get the height
    hsv_img = rgb2hsv(my_image) #turn RGB to hue, saturation, value

    #split matrix into three rows
    hue_img = hsv_img[:, :, 0]
    saturation_img = hsv_img[:,:, 1]
    value_img = hsv_img[:, :, 2]

    # get mean values from columns (brightness = mean value)
    mean_hue = np.mean(hue_img, axis=(0,1))
    mean_saturation = np.mean(saturation_img, axis=(0,1))
    mean_brightness = np.mean(value_img, axis=(0,1))

    #place values in output vector
    img_stats[2] = mean_hue
    img_stats[3] = mean_saturation
    img_stats[4] = mean_brightness

    return img_stats

#print(img_stats)
```

We want to calculate these stats for each of Rothko's paintings and store them in a pandas dataframe for plotting and analysis. Write code (using `calc_stats()` from above) to:

- Iterate over Rothko's paintings
- Compute these values for each image
- Add to a dataframe
- And write to disk as a csv (`mark-rothko.csv`).

In [3]:
```
from skimage import io
import skimage
import os
import pandas as pd

#use num_paintings variable

#use variable to keep track of row for csv file
csv_idx = 0;

w = np.zeros((num_paintings));
h = np.zeros((num_paintings));
mh = np.zeros((num_paintings));
ms = np.zeros((num_paintings));
mb = np.zeros((num_paintings));

for file in os.listdir('../data/mark-rothko/'):

    filename = os.path.join('../data/mark-rothko/', file)
    print(filename)
    #current_img = io.imread(filename)

    #use function to read in file and calculate stats
    img_stats = calc_stats(filename)

    #put img_stats into corresponding vectors (could be cleaner)
    w[csv_idx] = img_stats[0]
    h[csv_idx] = img_stats[1]
    mh[csv_idx] = img_stats[2]
    ms[csv_idx] = img_stats[3]
    mb[csv_idx] = img_stats[4]

    #increment index after placing array in csv matrix
    csv_idx = csv_idx + 1;

# print(w)

d = {'width': w, 'height': h, 'mean_hue': mh, 'mean_sat': ms, 'mean_br
ight': mb}
```

```
df = pd.DataFrame(data=d)

# w = w.transpose()
# h = h.transpose()
# mh = mh.transpose()
# ms = ms.transpose()
# mb = mb.transpose()
# df = pd.DataFrame(np.array(w, h, mh, ms, mb), columns=['w','h','mh',
'ms','mb'])

#not sure about the flags but this does work...
df.to_csv(r'../data/mark-rothko/export_dataframe.csv', index = False,
header = True)
```

```
../data/mark-rothko/yellow-cherry-orange.jpg
../data/mark-rothko/ochre-and-red-on-red-1.jpg
../data/mark-rothko/untitled-1968-1.jpg
../data/mark-rothko/no-1-untitled.jpg
../data/mark-rothko/untitled-7.jpg

/anaconda3/lib/python3.7/site-packages/skimage/color/colorconv.py:27
5: RuntimeWarning: divide by zero encountered in true_divide
  out[idx, 0] = (arr[idx, 1] - arr[idx, 2]) / delta[idx]
/anaconda3/lib/python3.7/site-packages/skimage/color/colorconv.py:28
3: RuntimeWarning: divide by zero encountered in true_divide
  out[idx, 0] = 4. + (arr[idx, 0] - arr[idx, 1]) / delta[idx]

../data/mark-rothko/not_detected_242137.jpg

/anaconda3/lib/python3.7/site-packages/skimage/color/colorconv.py:26
9: RuntimeWarning: divide by zero encountered in true_divide
  out_s = delta / out_v
/anaconda3/lib/python3.7/site-packages/skimage/color/colorconv.py:27
9: RuntimeWarning: divide by zero encountered in true_divide
  out[idx, 0] = 2. + (arr[idx, 2] - arr[idx, 0]) / delta[idx]
```

```
../data/mark-rothko/cat-newyork.jpg
../data/mark-rothko/not_detected_242122.jpg
../data/mark-rothko/not_detected_242136.jpg
../data/mark-rothko/no-37-no-19-slate-blue-and-brown-on-plum-1958.jp
g
../data/mark-rothko/untitled-6(1).jpg
../data/mark-rothko/untitled-1968-2.jpg
../data/mark-rothko/ochre-and-red-on-red-2.jpg
../data/mark-rothko/black-on-maroon.jpg
../data/mark-rothko/interior.jpg
../data/mark-rothko/not_detected_242120.jpg
../data/mark-rothko/red-white-and-brown.jpg
../data/mark-rothko/not_detected_242135.jpg
../data/mark-rothko/not_detected_242121.jpg
../data/mark-rothko/untitled-brown-and-gray.jpg
../data/mark-rothko/untitled-5.jpg
../data/mark-rothko/blue-and-gray.jpg
../data/mark-rothko/untitled-1.jpg
../data/mark-rothko/untitled-1948-2.jpg
../data/mark-rothko/not_detected_242119.jpg
../data/mark-rothko/not_detected_242125.jpg
../data/mark-rothko/not_detected_242131.jpg
../data/mark-rothko/not_detected_242124.jpg
../data/mark-rothko/not_detected_242118.jpg
../data/mark-rothko/sacrifice-of-iphigenia.jpg
../data/mark-rothko/untitled-1969-1(1).jpg
../data/mark-rothko/untitled-red-blue-orange-1955.jpg
../data/mark-rothko/untitled-2.jpg
../data/mark-rothko/export_dataframe.csv

---------------------------------------------------------------------
-------
OSError                                   Traceback (most recent cal
l last)
<ipython-input-3-a2f52e7a222c> in <module>
     22
     23     #use function to read in file and calculate stats
---> 24     img_stats = calc_stats(filename)
     25
     26     #put img_stats into corresponding vectors (could be clea
ner)

<ipython-input-2-324329d73ec5> in calc_stats(filename)
     17     img_stats = np.zeros((5, 1))
     18
---> 19     my_image = io.imread(filename) #read the image
     20     img_stats[0] = my_image.shape[0] #get the width
     21     img_stats[1] = my_image.shape[1] #get the height

/anaconda3/lib/python3.7/site-packages/skimage/io/_io.py in imread(f
```

```
name, as_gray, plugin, flatten, **plugin_args)
     60
     61     with file_or_url_context(fname) as fname:
---> 62         img = call_plugin('imread', fname, plugin=plugin, **
plugin_args)
     63
     64     if not hasattr(img, 'ndim'):

/anaconda3/lib/python3.7/site-packages/skimage/io/manage_plugins.py
in call_plugin(kind, *args, **kwargs)
    212                                 (plugin, kind))
    213
--> 214     return func(*args, **kwargs)
    215
    216


/anaconda3/lib/python3.7/site-packages/skimage/io/_plugins/pil_plugi
n.py in imread(fname, dtype, img_num, **kwargs)
     34     if isinstance(fname, string_types):
     35         with open(fname, 'rb') as f:
---> 36             im = Image.open(f)
     37             return pil_to_ndarray(im, dtype=dtype, img_num=i
mg_num)
     38     else:

/anaconda3/lib/python3.7/site-packages/PIL/Image.py in open(fp, mode
)
   2655         warnings.warn(message)
   2656     raise IOError("cannot identify image file %r"
-> 2657                   % (filename if filename else fp))
   2658
   2659 #

OSError: cannot identify image file <_io.BufferedReader name='../dat
a/mark-rothko/export_dataframe.csv'>
```

1C. Plotting Results

(15 points)

For this section we will create some simple plots with matplot lib showing distributions of image stats (mean value, hue, saturation, and resolution). Then we will produce large bitmap plots similar to Manovich's work above.

(see example notebooks for plotting)

In [ ]:
```
%matplotlib inline
```

P1. Distribution of sizes

First plot a histogram of image resolution using matplotlib and display inline.

In [ ]:
```
# import
%matplotlib inline

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

#read the csv file using pandas
df = pd.read_csv('../data/mark-rothko/export_dataframe.csv')
w = df[['width']]
w = w.values
h = df[['height']]
h = h.values

# print(w)
# print(h)

#use single value as resolution
#width times height

#vector for our resolutions
resolutions = np.zeros(w.size)

for i in range(0, w.size-1):
    resolutions[i] = w[i] * h[i]

num_bins = 100

n, bins, patches = plt.hist(resolutions, num_bins, facecolor='blue', a
lpha=0.5)#
#plt.show()

# print(n)
# print(bins)
# print(patches)

plt.xlabel('Num Pixels')
plt.ylabel('Num Paintings')
plt.title(r'Histogram Resolution')
# plt.plot(n, bins)
plt.show()
```

P2-P4. Distribution of Mean Hue, Saturation, Value

Next plot histograms of mean hue, saturation, and value, and dislpay inline below

In [ ]:
```
# your code for mean hue histogram

# import (not sure if I have to again)
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

#read the csv file using pandas (not sure if i need again)
df = pd.read_csv('../data/mark-rothko/export_dataframe.csv')
h = df[['mean_hue']]
hue = h.values

# print(w)
# print(h)

num_bins = 100

n, bins, patches = plt.hist(hue, num_bins, facecolor='blue', alpha=0.5
)#
#plt.show()

# print(n)
# print(bins)
# print(patches)

plt.xlabel('Mean Hue')
plt.ylabel('Num Paintings')
plt.title(r'Histogram Mean Hue')
# plt.plot(n, bins)
plt.show()
```

In [ ]:
```
# your code for mean saturation histogram

# import (not sure if I have to again)
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

#read the csv file using pandas (not sure if i need again)
df = pd.read_csv('../data/mark-rothko/export_dataframe.csv')
s = df[['mean_sat']]
sat = s.values

# print(w)
# print(h)

num_bins = 100

n, bins, patches = plt.hist(sat, num_bins, facecolor='blue', alpha=0.5
)#
#plt.show()

# print(n)
# print(bins)
# print(patches)

plt.xlabel('Mean Sat')
plt.ylabel('Num Paintings')
plt.title(r'Histogram Mean Saturation')
# plt.plot(n, bins)
plt.show()
```

In [ ]:
```
# your code for mean value histogram

# import (not sure if I have to again)
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

#read the csv file using pandas (not sure if i need again)
df = pd.read_csv('../data/mark-rothko/export_dataframe.csv')
b = df[['mean_bright']]
bright = b.values

# print(w)
# print(h)

num_bins = 100

n, bins, patches = plt.hist(bright, num_bins, facecolor='blue', alpha=
0.5)#
#plt.show()

# print(n)
# print(bins)
# print(patches)

plt.xlabel('Mean Brightness')
plt.ylabel('Num Paintings')
plt.title(r'Histogram Mean Brightness')
# plt.plot(n, bins)
plt.show()
```

P5. Scatterplot with matplotlib (mean value vs. mean hue)

Now produce a simple scatter plot of mean value against mean hue.

(see example notebook on plotting)

In [ ]:
```
# your code for scatter plot of mean_value (X) against mean_hue (Y)

# import (not sure if I have to again)
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

#read the csv file using pandas (not sure if i need again)
df = pd.read_csv('../data/mark-rothko/export_dataframe.csv')
b = df[['mean_bright']]
h = df[['mean_hue']]
bright = b.values
hue = h.values

# print(w)
# print(h)

num_bins = 100

#n, bins, patches = plt.hist(bright, num_bins, facecolor='blue', alpha
=0.5)#
#plt.show()

plt.scatter(bright, hue, marker='o');

# print(n)
# print(bins)
# print(patches)

plt.xlabel('Brightness')
plt.ylabel('Hue')
plt.title(r'Scatter Plots Brightness v Hue')
# plt.plot(n, bins)
plt.show()
```

P6-P7. Produce Large Bitmap Figures illustrating your results

(see example notebook on producing large tiled image figures: ../examples/large_figures.ipynb (../examples/large_figures.ipynb))

In [ ]:
```
# from skimage import io
from PIL import Image
import matplotlib.pyplot as plt
```

Step 1: Generate thumbnails from full-resolution scraped images

Write a `make_thumbnail()` function that takes a filename, imagepath, and thumbnail path as arguments

In [ ]:
```
import glob, os

#path = ../data/mark-rothko/

def make_thumbnail(filename, imagepath, thumbnail_path):
    full_path = imagepath + filename; #concatenate strings
    my_image = Image.open(full_path)
    size = 128, 128
    my_image.thumbnail(size)
    my_image.save(thumbnail_path + filename)
```

Create a folder to store your thumbnails

In [ ]:
```
import os

dir_name = 'thumb_rothko'

try:
    os.mkdir(dir_name)
except FileExistsError:
    print("Directory already exists")
```

Iterate over your Rothko paintings and write thumbnails to disk

In [ ]:
```
for file in os.listdir('../data/mark-rothko/'):

    #filename = os.path.join('../data/mark-rothko/', file)
    #print(filename)
    #current_img = io.imread(filename)
    if file.endswith(".jpg"):
        make_thumbnail(file, '../data/mark-rothko/', 'thumb_')
    else:
        continue
```

Step 2: Create large plots on an empty bitmap canvas, placing thumbnails based on feature coordinates.

Make a folder to save result (`../data/mark-rothko/results`)

In [ ]:
```
# importing os module
import os

# Directory
directory = "results"

# Parent Directory path
parent_dir = "../data/mark-rothko/"

# Path
path = os.path.join(parent_dir, directory)

# Create the directory
# 'GeeksForGeeks' in
# '/home / User / Documents'
os.mkdir(path)
print("Directory '% s' created" % directory)
```

Plot mean value vs. mean hue with image thumbnails on large bitmap

In [ ]:
```
import pandas as pd
from PIL import Image
import os

# create background image
GLOBAL_WIDTH = 7500
bg_color = (192, 192, 192) # gray, you can choose your own
figure = Image.new('RGB', (GLOBAL_WIDTH, GLOBAL_WIDTH), bg_color)

#read the csv file using pandas
df = pd.read_csv('../data/mark-rothko/export_dataframe.csv')
b = df[['mean_bright']]
b = b.values
h = df[['mean_hue']]
h = h.values

len_list= h.size

# i think there is something wrong because i have the data but
# I dont know which values belong to which painting
# i will assume that the file reader reads the data in the same order
# every time

idx = 0;

for file in os.listdir('.'):
    if file.endswith(".jpg"):
        thumb_img = Image.open(file)
        figure.paste(thumb_img, (b[idx]*GLOBAL_WIDTH, h[idx]*GLOBAL_WI
DTH))
        idx = idx+1
        print(b[idx]*GLOBAL_WIDTH)
    else:
        continue

figure.save("b_h_bitmap.jpg")

# not really sure I did it right ...
# it is definitely messy, the files are all over the place
# I can definitely fix that.
```

Produce a second plot: mean value vs mean saturation

In [ ]:
```
import pandas as pd
from PIL import Image
import os

# create background image
GLOBAL_WIDTH = 7500
bg_color = (192, 192, 192) # gray, you can choose your own
figure = Image.new('RGB', (GLOBAL_WIDTH, GLOBAL_WIDTH), bg_color)

print("Hello")

#read the csv file using pandas
df = pd.read_csv('../data/mark-rothko/export_dataframe.csv')
b = df[['mean_bright']]
b = b.values
s = df[['mean_sat']]
s = s.values

len_list= s.size

# i think there is something wrong because i have the data but
# I dont know which values belong to which painting
# i will assume that the file reader reads the data in the same order
# every time

idx = 0;

for file in os.listdir('.'):
    if file.endswith(".jpg"):
        thumb_img = Image.open(file)
        figure.paste(thumb_img, (b[idx]*GLOBAL_WIDTH, s[idx]*GLOBAL_WI
DTH))
        idx = idx+1
        print(b[idx]*GLOBAL_WIDTH)
    else:
        continue

figure.save("b_s_bitmap.jpg")

# not really sure I did it right ...
# it is definitely messy, the files are all over the place
# I can definitely fix that.
```

Display the figures inline in this notebook

In [ ]:
```
# your code here
```

Part 2: Extension

(70 points total)

For this part, you will repeat the above image feature summary analysis (mean brightness, mean hue) using a dataset of your choice. Your data should have approximately n <= 100 images. Your output should be a similar tiled image as produced in the previous section, along with a short paragraph describing your results and why they are interesting.

2A. Scraping/downloading your new imagery

(10 points)

In [ ]:
```
# import libraries
from bs4 import BeautifulSoup
import os
import requests

#set up data paths
DATA_DIR = '../data/'
ARTIST_URL = 'https://www.wikiart.org/en/{artist}/all-works/text-list'
PAINTING_URL = 'https://www.wikiart.org{painting_path}'

#make folder for paintings we want to download
if not os.path.exists(DATA_DIR):
    os.makedirs(DATA_DIR)

#get list of paintings
artist_name = 'jackson-pollock'
url_query = ARTIST_URL.format(artist=artist_name)
artist_page = requests.get(url_query)

# check for request error
try:
    artist_page.raise_for_status()
except requests.exceptions.HTTPError as e:
    print("Error trying to retrieve {}".format(artist_page.url))
    raise e

#call web scrapper
soup = BeautifulSoup(artist_page.text, 'lxml')

#create image storage directory for artist if it doesn't exist
IMAGE_DIR = os.path.join(DATA_DIR, artist_name)
```

```
if not os.path.exists(IMAGE_DIR):
    os.makedirs(IMAGE_DIR)

#make array for paths
painting_paths = []

# retreive all rows in painting-list
for li in soup.find_all('li', {'class': 'painting-list-text-row'}):

    # retrieve all links in the current row
    for link in li.find_all('a'):
        href = link.get('href')
        # store in dictionary
        painting_paths.append(href)

print(len(painting_paths))
# painting_paths

#store number of paintings
num_paintings = len(painting_paths);

#download
def download_and_save(painting_url):
    r_painting_page = requests.get(painting_url)
    soup = BeautifulSoup(r_painting_page.text, 'lxml')
    for img in soup.find_all('img', {'class': 'ms-zoom-cursor'}):
        img_url = img['src']
        img_url = img_url.split('!')[0]
        filename = img_url.split('/')[-1]

        outfile = os.path.join(IMAGE_DIR, filename)
        if not os.path.exists(outfile):
            print("downloading {}: {}".format(filename, img_url))
            r = requests.get(img_url, outfile)
            with open(outfile, 'wb') as f:
                f.write(r.content)
        else:
            #print("skipping {}".format(filename))
            pass

#use our custom function to download paintings
for path in painting_paths:
    painting_path = PAINTING_URL.format(painting_path=path)
    download_and_save(painting_path)
```

```
86
downloading going-west.jpg: https://uploads6.wikiart.org/images/jack
son-pollock/going-west.jpg
downloading figures-in-a-landscape(1).jpg: https://uploads2.wikiart.
```

```
org/figures-in-a-landscape(1).jpg
downloading landscape-with-steer-1937(2).jpg: https://uploads6.wikia
rt.org/landscape-with-steer-1937(2).jpg
downloading landscape-with-steer-1937(3).jpg: https://uploads6.wikia
rt.org/landscape-with-steer-1937(3).jpg
downloading the-flame-1938(1).jpg: https://uploads3.wikiart.org/the-
flame-1938(1).jpg
downloading man-with-knife-1940(1).jpg: https://uploads0.wikiart.org
/man-with-knife-1940(1).jpg
downloading circle-1941(1).jpg: https://uploads0.wikiart.org/circle-
1941(1).jpg
downloading untitled-1941(2).jpg: https://uploads8.wikiart.org/untit
led-1941(2).jpg
downloading untitled-1941(3).jpg: https://uploads8.wikiart.org/untit
led-1941(3).jpg
downloading bird-1941(1).jpg: https://uploads8.wikiart.org/bird-1941
(1).jpg
downloading mask(1).jpg: https://uploads1.wikiart.org/mask(1).jpg
downloading sheet-of-studies-1941(1).jpg: https://uploads3.wikiart.o
rg/sheet-of-studies-1941(1).jpg
downloading birth(1).jpg: https://uploads3.wikiart.org/birth(1).jpg
downloading moon-woman-1942(1).jpg: https://uploads7.wikiart.org/moo
n-woman-1942(1).jpg
downloading animals-and-figures(1).jpg: https://uploads0.wikiart.org
/animals-and-figures(1).jpg
downloading male-and-female(1).jpg: https://uploads7.wikiart.org/mal
e-and-female(1).jpg
downloading stenographic-figure(1).jpg: https://uploads2.wikiart.org
/stenographic-figure(1).jpg
downloading untitled(4).jpg: https://uploads4.wikiart.org/untitled(4
).jpg
downloading the-she-wolf(1).jpg: https://uploads2.wikiart.org/the-sh
e-wolf(1).jpg
downloading blue-moby-dick(1).jpg: https://uploads6.wikiart.org/blue
-moby-dick(1).jpg
downloading composition-with-pouring-ii(1).jpg: https://uploads3.wik
iart.org/composition-with-pouring-ii(1).jpg
downloading untitled(5).jpg: https://uploads4.wikiart.org/untitled(5
).jpg
downloading mural.jpg: https://uploads1.wikiart.org/images/jackson-p
ollock/mural.jpg
downloading the-moon-woman-cuts-the-circle-1943.jpg: https://uploads
4.wikiart.org/images/jackson-pollock/the-moon-woman-cuts-the-circle-
1943.jpg
downloading untitled-1944(1).jpg: https://uploads0.wikiart.org/untit
led-1944(1).jpg
downloading totem-lesson-2-1945(1).jpg: https://uploads2.wikiart.org
/totem-lesson-2-1945(1).jpg
downloading pattern(1).jpg: https://uploads5.wikiart.org/pattern(1).
jpg
```

```
downloading untitled(3).jpg: https://uploads3.wikiart.org/untitled(3
).jpg
downloading eyes-in-the-heat-1946(2).jpg: https://uploads0.wikiart.o
rg/eyes-in-the-heat-1946(2).jpg
downloading shimmering-substance(1).jpg: https://uploads2.wikiart.or
g/shimmering-substance(1).jpg
downloading the-tea-cup(1).jpg: https://uploads1.wikiart.org/the-tea
-cup(1).jpg
downloading circumcision-january(1).jpg: https://uploads0.wikiart.or
g/circumcision-january(1).jpg
downloading the-key(1).jpg: https://uploads6.wikiart.org/the-key(1).
jpg
downloading pollock.jpg: https://uploads1.wikiart.org/00229/images/j
ackson-pollock/pollock.jpg
downloading alchemy-1947(2).jpg: https://uploads6.wikiart.org/alchem
y-1947(2).jpg
downloading full-fathom-five(1).jpg: https://uploads6.wikiart.org/fu
ll-fathom-five(1).jpg
downloading galaxy-1947.jpg: https://uploads7.wikiart.org/images/jac
kson-pollock/galaxy-1947.jpg
downloading enchanted-forest-1947.jpg: https://uploads7.wikiart.org/
images/jackson-pollock/enchanted-forest-1947.jpg
downloading lucifer-1947.jpg: https://uploads8.wikiart.org/images/ja
ckson-pollock/lucifer-1947.jpg
downloading cathedral-1947.jpg: https://uploads1.wikiart.org/images/
jackson-pollock/cathedral-1947.jpg
downloading reflections-of-the-big-dipper-1947.jpg: https://uploads2
.wikiart.org/images/jackson-pollock/reflections-of-the-big-dipper-19
47.jpg
downloading untitled-o-connor-thaw-770(1).jpg: https://uploads3.wiki
art.org/untitled-o-connor-thaw-770(1).jpg
downloading untitled-o-connor-thaw-771(1).jpg: https://uploads3.wiki
art.org/untitled-o-connor-thaw-771(1).jpg
downloading no-1-1948(1).jpg: https://uploads3.wikiart.org/no-1-1948
(1).jpg
downloading number-23(1).jpg: https://uploads6.wikiart.org/number-23
(1).jpg
downloading number-3(3).jpg: https://uploads6.wikiart.org/number-3(3
).jpg
downloading dc30110d-dc5a-40cd-9345-d49e74059b13.jpg: https://upload
s5.wikiart.org/temp/dc30110d-dc5a-40cd-9345-d49e74059b13.jpg
downloading number-4-gray-and-red-1948.jpg: https://uploads3.wikiart
.org/images/jackson-pollock/number-4-gray-and-red-1948.jpg
downloading number-13a-arabesque-1948.jpg: https://uploads2.wikiart.
org/images/jackson-pollock/number-13a-arabesque-1948.jpg
downloading summertime-number-9a-1948.jpg: https://uploads3.wikiart.
org/images/jackson-pollock/summertime-number-9a-1948.jpg
downloading composition-white-black-blue-and-red-on-white-1948.jpg:
https://uploads0.wikiart.org/images/jackson-pollock/composition-whit
e-black-blue-and-red-on-white-1948.jpg
```

```
downloading number-6-1949(1).jpg: https://uploads1.wikiart.org/numbe
r-6-1949(1).jpg
downloading number-1(1).jpg: https://uploads5.wikiart.org/number-1(1
).jpg
downloading number-8-detail(1).jpg: https://uploads4.wikiart.org/num
ber-8-detail(1).jpg
downloading number-3(2).jpg: https://uploads6.wikiart.org/number-3(2
).jpg
downloading number-7-out-of-the-web-1949.jpg: https://uploads6.wikia
rt.org/images/jackson-pollock/number-7-out-of-the-web-1949.jpg
downloading number-10-1949.jpg: https://uploads1.wikiart.org/images/
jackson-pollock/number-10-1949.jpg
downloading number-17-1949.jpg: https://uploads0.wikiart.org/images/
jackson-pollock/number-17-1949.jpg
downloading number-12-1949.jpg: https://uploads0.wikiart.org/images/
jackson-pollock/number-12-1949.jpg
```

2B. Calculating image features

(10 points)

Model your features on the above exercise, or incorporate other stats (variance, edge count, etc.)

In [ ]:
```
from skimage import io
import skimage
import os
import pandas as pd

#use num_paintings variable

#use variable to keep track of row for csv file
csv_idx = 0;

w = np.zeros((num_paintings));
h = np.zeros((num_paintings));
mh = np.zeros((num_paintings));
ms = np.zeros((num_paintings));
mb = np.zeros((num_paintings));

for file in os.listdir('../data/jackson-pollock/'):

    filename = os.path.join('../data/jackson-pollock/', file)
    print(filename)
    #current_img = io.imread(filename)

    #use function to read in file and calculate stats
    img_stats = calc_stats(filename)
```

```
    #put img_stats into corresponding vectors (could be cleaner)
    w[csv_idx] = img_stats[0]
    h[csv_idx] = img_stats[1]
    mh[csv_idx] = img_stats[2]
    ms[csv_idx] = img_stats[3]
    mb[csv_idx] = img_stats[4]

    #increment index after placing array in csv matrix
    csv_idx = csv_idx + 1;

# print(w)

d = {'width': w, 'height': h, 'mean_hue': mh, 'mean_sat': ms, 'mean_br
ight': mb}
df = pd.DataFrame(data=d)

# w = w.transpose()
# h = h.transpose()
# mh = mh.transpose()
# ms = ms.transpose()
# mb = mb.transpose()
# df = pd.DataFrame(np.array(w, h, mh, ms, mb), columns=['w','h','mh',
'ms','mb'])

#not sure about the flags but this does work...
df.to_csv(r'../data/jackson-pollock/export_dataframe.csv', index = Fal
se, header = True)
```

2C. Produce and Display output plots (results)

(25 points)

Produce high resolution results images, and display them inline in the notebook

In [ ]:
```
# your code here

#I am not going to copy paste code here since I dont think it actually
works for the bitmaps. i am not sure what is wrong.
#all the other plots worked ok.
```

2D. Describe your Results

(25 points)

Replace the contents of the markdown cell below with a two paragraph summary of your extension work.

To be honest I didnt get to really finish and I am not sure I did it right. I am just going to most possible points. I am new to jupyter so it will take a bit to get used to it. not really sure how the namespace thing works. i miss matlab.

i think the picture i got turned out ok.

this is an intersting process and i am excited to do it again with other data, maybe sound based on mfcc?

In [ ]:

References

Additional Cultural Archives:

- The Getty (https://www.getty.edu/art/collection/) (The J. Paul Getty Museum, LA)
- The Met Collection (https://www.metmuseum.org/art/collection) (Metropolitan Museum of Art, NYC)
- MoMA (Museum of Modern Art) online collection: https://www.moma.org/collection/ (https://www.moma.org/collection/)
  - Our evolving collection contains almost 200,000 works of modern and contemporary art. More than 85,000 works are currently available online.
- Metropolitan Museum of Art collection on Archive.org: https://archive.org/details/metropolitanmuseumofart-gallery (https://archive.org/details/metropolitanmuseumofart-gallery?&sort=-downloads&page=2)
- MoMA exhibition images (https://www.moma.org/collection/) (showing how paintings were installed)
  - read about it here You Can Now Explore Every MoMA Exhibit Since 1929 for Free Online (https://mymodernmet.com/museum-of-modern-art-exhibition-history/?fbclid=IwAR3LkAPAXmDJ4C9zJn6ujfmhh2zNp6GJL9ysHTMgoKPS5ARp8jx3EklalUk)
- Paul Klee notebooks (http://www.kleegestaltungslehre.zpk.org/ee/ZPK/BF/2012/01/01/001/)
  - read about it here (http://www.openculture.com/2016/03/3900-pages-of-paul-klees-personal-notebooks-are-now-online.html?fbclid=IwAR1_dGLxqy0YAiGuxJD2uTVUiyS0sSJuoX8iKuy_k01LWHbAYcbprNp4hd4)

In [ ]: