

stat29000project02solutions

February 5, 2020

1 STAT29000 Project 2 Solutions

1.1 Question 1

```
[15]: import io
import requests
import numpy as np
import pandas as pd
from PIL import Image
from scipy import stats
import backoff
from matplotlib import pyplot as plt
```

```
[4]: sample = np.random.normal(4, np.sqrt(.06), 100000)
print(sample.mean())
print(sample.var())
```

```
4.000411636035883
0.05991908024096071
```

```
[5]: sample = np.random.gamma(266.6666, .015, 100000)
print(sample.mean())
print(sample.var())
```

```
3.998148036197811
0.06038514243747886
```

1.2 Question 2

```
[6]: data = pd.read_csv("https://raw.githubusercontent.com/zygmuntz/goodbooks-10k/
↪master/books.csv")
data.shape
```

```
[6]: (10000, 23)
```

```
[ ]: # Since we are scraping a live website, let's not overwhelm the website and get
      ↪blocked. We will
      # use exponential backoff when scraping webpages.
      import backoff
      @backoff.on_exception(backoff.expo, requests.exceptions.RequestException)
      def get_url(url):
          return requests.get(url)

      images = []
      for index, row in data.iterrows():
          img_response = get_url(f'{row["image_url"]}\'')
          image_file = io.BytesIO(img_response.content)
          img = Image.open(image_file)
          images.append(img)

      data['cover_images'] = images
      # data.to_pickle("books_with_images")
```

```
[7]: data = pd.read_pickle("~/Dropbox/work/data_mine/spring2020/stat29000/project02/
      ↪books_with_images_new")

      def is_warm(r,g,b):
          return r > g or (r >= 128 and g > b)

      percent_warm = []
      for index, row in data.iterrows():
          width, height = row['cover_images'].size
          rgbimg = row['cover_images'].convert("RGB")
          warm = []
          for i in range(width):
              for j in range(height):
                  r,g,b = rgbimg.getpixel((i, j))
                  warm.append(is_warm(r,g,b))
          percent_warm.append(sum(warm)/len(warm))

      data['percent_warm'] = percent_warm
      data['percent_cool'] = 1 - data['percent_warm']
```

```
[9]: warmest = data.loc[data['percent_warm'] == data['percent_warm'].max()]
      coolest = data.loc[data['percent_warm'] == data['percent_warm'].min()]
      print(len(coolest))
      print(len(warmest))
```

28
3428

```
[16]: for idx, book in warmest.sample(5).iterrows():  
      plt.figure()  
      plt.imshow(book['cover_images'])
```







```
[17]: # an example of the default image is
default_image = warmest['cover_images'].iloc[1]

# remove all rows where the default image is used
replace_with_data = data.loc[data['cover_images'] != default_image]

# or
data.drop(data.loc[data['cover_images'] == default_image].index, inplace=True)

len(data)
```

[17]: 6668

```
[18]: middling = data.loc[data['percent_warm'].between(0.49, 0.51)]
print(f'Length: {len(middling)}\n')
for idx, book in enumerate(middling['original_title'].sort_values()):
    if idx < 5:
        print(book)
```

Length: 106

1Q84 Book 1 [Ichi-kyū-hachi-yon]
A People's History of the United States: 1492 to Present
Absolute Batman Hush
Alienated
All the Ugly and Wonderful Things

```
[19]: print(stats.spearmanr(data['percent_warm'], data['average_rating']))  
data['average_rating'].plot.kde()
```

SpearmanrResult(correlation=-0.02765985051964235, pvalue=0.02390505118404748)

```
[19]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc0f3b0e748>
```

