**UGCA F2018 Assignment 4 (Group work)**

**Date Handed out: 6th November 2018**

**Date Due: 19th November 2018 by 11:59 p.m.**

**Is a Picture Worth a Thousand Words?**

The file **Instagram\_zara.csv** has data on images posted by Zara on its Instagram page. While the initial objective of this assignment was to make you scrape images and perform image analytics, Instagram changed its HTML design over the weekend, rendering my scraper script useless (we are working on changing the script). This has forced me to use some data I had scraped before the change.

Each row of the data file contains:

1. An image ID (a number, not necessarily in sequence)
2. post\_caption (description of the image provided by Zara)
3. #\_comments (number of comments posted for an image)
4. #\_likes (number of likes for an image)
5. Image\_labels (words describing the content of an image, obtained from Google Vision analytics)

**Task A.** Create a metric (score) for **engagement** by using a weighted sum of #\_likes and #\_comments. However, first normalize #\_likes and #\_comments such that they both have values between 0 and 1. You can normalize the #\_likes by dividing by the maximum #\_likes in the column in your data and do the same for #\_comments, so that #\_likes and comments will be in the range [0,1]. For example, if the maximum number of likes for a post was 2 million in the data set, divide every cell in the #\_likes column by 2 million, so that the maximum value in the column will be 1. Do the same for the #\_comments column. Now create an engagement score = .4\*#\_likes (normalized) + .6\*#\_comments (normalized). Create a column called **binary** (lowercase only) where value =1 (stands for high engagement) or 0 (stands for low engagement) based on whether the engagement score is above or below the median value.

**Task B.** Run a logistic regression with **binary** as the dependent variable, and the image\_labels as independent variables. Before running the regression, replace image\_label with **text**, since the script expects text to be the name for the column containing text. What is the accuracy (show the confusion matrix) of this prediction model? The idea is to be able to predict the engagement level for an image.

Accuracy = 1 - # prediction errors / total # cases

What accuracy do you get by using the post\_caption words as the independent variables instead of image\_labels? Finally, what accuracy do you get by combining (concatenating) the image\_labels and post\_caption and using them together as independent variables? What can you conclude from your analysis?

**Note: Do a word frequency analysis and word replacement on the image\_labels as well as post\_captions before running the logistic regression; it will most likely increase the accuracy of predictions.**

Task C. Perform topic modeling (LDA) on the original image\_labels (do not use any word replacements here). Choose an appropriate number of topics. You may want to start with 5 topics, but adjust the number up or down depending on the word distributions you get. **Decide on suitable names for each topic.**

Now sort the data from high to low engagement score (don’t use the binary column, use the actual engagement score instead), and take the highest and the lowest quartiles (by engagement score). What are the main differences in the **average** topic weights of images across the two quartiles (e.g., greater weight of some topics in the highest versus lowest engagement quartiles)? Show the main results in a table.

Task D. What advice would you give National Geographic if it wants to increase engagement on its Instagram page based on your findings in Tasks B and C?

**Deliverables:** A word or pdf file with tables and answers. Write the names of all team members **inside** the document.

**Guidelines**

For logistic regression, use **logistic\_new.py**. This script expects a two-column file as input: an output column **binary** (lowercase) and a text column called **text**. Make sure **binary** is the first column and **text** the second, The name of the input data file (csv format) should be provided in line 159. The script prompts you whether the data has text. Respond with **Yes**. The output file (containing the confusion matrix) is Confusion.png. The accuracy can be seen in the confusion matrix file.

For topic modeling (LDA) use the script **lda\_reviews\_new.py**. This script works on a 2-column file yelp\_reviews.xslx with column names Restaurant\_name and Restaurant\_review. The script prompts you for the names of the two columns as well as the number of topics to be created. For your convenience, I have included the Yelp file. The script produces two output files: topic\_word\_dist.xlsx (which contains the weights of words for each topic) and restaurant\_topic\_dist.xlsx (the weights of topics for each restaurant).