A2 - Bias in Data Assignment

The goal of this assignment is to explore the concept of bias through data on Wikipedia articles - specifically, articles on political figures from a variety of countries. For this assignment, you will combine a dataset of Wikipedia articles with a dataset of country populations, and use a machine learning service called ORES to estimate the quality of each article.

You are expected to perform an analysis of how the coverage of politicians on Wikipedia and the quality of articles about politicians varies between countries. Your analysis will consist of a series of tables that show:

1. the countries with the greatest and least coverage of politicians on Wikipedia compared to their population.
2. the countries with the highest and lowest proportion of high quality articles about politicians.
3. a ranking of geographic regions by articles-per-person and proportion of high quality articles.

You are also expected to write a short reflection on the project that focuses on how both your findings from this analysis and the process you went through to reach those findings helps you understand the causes and consequences of biased data in large, complex data science projects.

Step 1: Getting the Article and Population Data

The first step is getting the data, which lives in several different places. The Wikipedia politicians by country dataset can be found on Figshare. Read through the documentation for this repository, then download and unzip it to extract the data file, which is called `page_data.csv`.

The population data is available in CSV format as `WPDS_2020_data.csv`. This dataset is drawn from the world population data sheet published by the Population Reference Bureau.

Step 2: Cleaning the Data

Both `page_data.csv` and `WPDS_2020_data.csv` contain some rows that you will need to filter out and/or ignore when you combine the datasets in the next step. In the case of `page_data.csv`, the dataset contains some page names that start with the string "`Template:`". These pages are not Wikipedia articles, and should not be included in your analysis.

Similarly, `WPDS_2020_data.csv` contains some rows that provide cumulative regional population counts, rather than country-level counts. These rows are distinguished by having ALL CAPS values in the 'geography' field (e.g. AFRICA, OCEANIA). These rows won't match the country values in `page_data.csv`, but you will want to retain them (either in the original file, or a separate file) so that you can report coverage and quality by region in the analysis section.

Step 3: Getting Article Quality Predictions

Now you need to get the predicted quality scores for each article in the Wikipedia dataset. We're using a machine learning system called ORES. This was originally an acronym for "Objective Revision Evaluation Service" but was simply renamed “ORES”. ORES is a machine learning tool that can provide estimates of Wikipedia article quality. The article quality estimates are, from best to worst:

1. FA - Featured article
2. GA - Good article
3. B - B-class article
4. C - C-class article
5. Start - Start-class article
6. Stub - Stub-class article

These were learned based on articles in Wikipedia that were peer-reviewed using the Wikipedia content assessment procedures.These quality classes are a sub-set of quality assessment categories developed by Wikipedia editors. For this assignment, you only need to know that these categories exist, and that ORES will assign one of these 6 categories to any `rev_id` you send it.

In order to get article predictions for each article in the Wikipedia dataset, you will first need to read `page_data.csv` into Python (or R), and then read through the dataset line by line, using the value of the `rev_id` column to make an API query.

You have two options for getting data from the ORES:

Option 1

Install and run the ORES client (Python only)

You can pip install ORES in your local notebook environment (https://github.com/wikimedia/ores installation instructions). This will allow you to get scores for lists of multiple `rev_id` values in a single batch -- you can even send all ~50k

articles in the `page_data.csv` in a single batch! Although, that might not be a good strategy. Here's some demo code:

```
from ores import api

#please provide this useragent string (second arg below) to help
the ORES team track requests

ores_session = api.Session("https://ores.wikimedia.org", "DATA 512
Class project <your_email_address@uw.edu>")

#where 1234, 5678, 91011 below are rev_ids...

results = ores_session.score("enwiki", ["articlequality"], [1234,
5678, 91011])

for score in results:

    print(score)

#where the value for 'prediction' in each response below
corresponds to the predicted article quality class

{'articlequality': {'score': {'prediction': 'B', 'probability':
{'GA': 0.005565225912988614, 'Stub': 0.285072978841463, 'C':
0.1237249061020009, 'B': 0.2910788689339172, 'Start':
0.2859984921969326, 'FA': 0.008559528012697881}}}}

{'articlequality': {'score': {'prediction': 'Start', 'probability':
{'GA': 0.005264197821210708, 'Stub': 0.40368617053424666, 'C':
0.021887833774629408, 'B': 0.029933164235917967, 'Start':
0.5352849001253548, 'FA': 0.0039437335086407645}}}}

{'articlequality': {'score': {'prediction': 'Stub', 'probability':
{'GA': 0.0033975128938096197, 'Stub': 0.8980284163392759, 'C':
0.012167869601103 09, 'B': 0.01579141569356552, 'Start':
0.06809640787450176, 'FA': 0.0025183775977442226}}}}
```

Option 2

Use the REST API endpoint (Python or R)

The ORES REST API is configured fairly similarly to the pageviews API we used for Assignment 1. You should review the ORES REST documentation. It expects a revision ID, which is the third column in the Wikipedia dataset, and a model, which is `"articlequality"`.

Whether you query the API or use the client, you will notice that ORES returns a `prediction` value that contains the name of one category, as well as `probability` values for each of the 6 quality categories. For this assignment, you only need to capture and use the value for `prediction`.

Note: It's possible that you will be unable to get a score for a particular article. If that happens, make sure to maintain a log of articles for which you were not able to retrieve an ORES score. This log can be saved as a separate file, or (if it's only a few articles), simply printed and logged within the notebook. The choice is up to you.

Step 3: Combining the Datasets

Some processing of the data will be necessary! In particular, you'll need to - after retrieving and including the ORES data for each article - merge the wikipedia data and population data together. Both have fields containing country names for just that purpose. After merging the data, you'll invariably run into entries which cannot be merged. Either the population dataset does not have an entry for the equivalent Wikipedia country, or vise versa.

Please remove any rows that do not have matching data, and output them to a CSV file called:

```
wp_wpds_countries-no_match.csv
```

Consolidate the remaining data into a single CSV file called:

```
wp_wpds_politicians_by_country.csv
```

The schema for that file should look something like this:

Column
country
article_name
revision_id
article_quality_est.
population

Note: revision_id here is the same thing as `rev_id`, which you used to get scores from ORES.

Step 4: Analysis

Your analysis will consist of calculating the proportion (as a percentage) of articles-per-population and high-quality articles for each country AND for each geographic region. By "high quality" articles, in this case we mean the number of articles about politicians in a given country that ORES predicted would be in either the "FA" (featured article) or "GA" (good article) classes.

Examples:

- if a country has a population of 10,000 people, and you found 10 FA or GA class articles about politicians from that country, then the percentage of articles-per-population would be .1%.
- if a country has 10 articles about politicians, and 2 of them are FA or GA class articles, then the percentage of high-quality articles would be 20%.

Step 5: Results

Your results from this analysis will be published in the form of data tables. You are being asked to produce six total tables, that show:

1. Top 10 countries by coverage: 10 highest-ranked countries in terms of number of politician articles as a proportion of country population
2. Bottom 10 countries by coverage: 10 lowest-ranked countries in terms of number of politician articles as a proportion of country population
3. Top 10 countries by relative quality: 10 highest-ranked countries in terms of the relative proportion of politician articles that are of GA and FA-quality
4. Bottom 10 countries by relative quality: 10 lowest-ranked countries in terms of the relative proportion of politician articles that are of GA and FA-quality
5. Geographic regions by coverage: Ranking of geographic regions (in descending order) in terms of the total count of politician articles from countries in each region as a proportion of total regional population
6. Geographic regions by coverage: Ranking of geographic regions (in descending order) in terms of the relative proportion of politician articles from countries in each region that are of GA and FA-quality

Embed these tables in your Jupyter notebook. You do not need to graph or otherwise visualize the data for this assignment, although you are welcome to do so in addition to generating the data tables described above, if you wish.

Reminder: you will find the list of geographic regions, which countries are in each region, and total regional population in the raw `WPDS_2020_data.csv` file. See "Step 2: Cleaning the data" above for more information.

Writeup: Reflections and Implications

Write a few paragraphs, either in the README or at the end of the notebook, reflecting on what you have learned, what you found, what (if anything) surprised you about your findings, and/or what theories you have about why any biases might exist (if you find they exist). You can also include any questions this assignment raised for you about bias, Wikipedia, or machine learning.

In addition to any reflections you want to share about the process of the assignment, please respond (briefly) to **at least three** of the questions below:

1. What biases did you expect to find in the data (before you started working with it), and why?
2. What (potential) sources of bias did you discover in the course of your data processing and analysis?
3. What might your results suggest about (English) Wikipedia as a data source?
4. What might your results suggest about the internet and global society in general?
5. Can you think of a realistic data science research situation where using these data (to train a model, perform a hypothesis-driven research, or make business decisions) might create biased or misleading results, due to the inherent gaps and limitations of the data?
6. Can you think of a realistic data science research situation where using these data (to train a model, perform a hypothesis-driven research, or make business decisions) might still be appropriate and useful, despite its inherent limitations and biases?
7. How might a researcher supplement or transform this dataset to potentially correct for the limitations/biases you observed?

This section doesn't need to be particularly long or thorough, but we'll expect you to write at least a couple paragraphs.

Submission instructions

1. Complete your analysis and write up
2. Check all deliverables into your GitHub repo
3. Submit the link to your GitHub repo through the Assignment 2 submission form on Canvas

Required deliverables

A directory in your GitHub repository called `data-512-a2` that contains at minimum the following files:

1. your two source data files and a description of each
2. 1 final data file in CSV format that contains all articles you analyzed, the corresponding country and population, and their predicted quality score.
3. 1 Jupyter notebook named `hcds-a2-bias` that contains all code as well as information necessary to understand each programming step, as well your findings (six tables) and your writeup (if you have not included it in the README).
4. 1 README file in .txt or .md format that contains information to reproduce the analysis, including data descriptions, attributions and provenance information, and descriptions of all relevant resources and documentation (inside and outside the repo) and hyperlinks to those resources, and your writeup (if you have not included it in the notebook).
5. 1 LICENSE file that contains an MIT LICENSE for your code.

If you created any additional process or incremental files in the course of your data processing and analysis (for example, a list of articles for which you were not able to gather ORES scores), please include these in the folder as well, and briefly describe them in the README.

Helpful tips

Read all instructions carefully before you begin

Read all API documentation carefully before you begin

Experiment with queries in the sandbox of the technical documentation for the API to familiarize yourself with the schema and the data

Explore the data a bit before starting to be sure you understand how it is structured and what it contains

Ask questions on Slack if you're unsure about anything. If you need more help, come to office hours or schedule a time to meet with Yihan or Jonathan.

When documenting/describing your project, think: "If I found this GitHub repo, and wanted to fully reproduce the analysis, what information would I want? What information would I need?"