Predicting Off-Topic Questions on StackOverflow

Hey there! So, for a variety of privacy and legal reasons we can't really give you *our* data to work with, but the folks at StackExchange have been so kind as to provide *their* data under a creative commons license. So, this challenge is going to be about StackOverflow!

Motivation

On StackOverflow, users post questions about programming problems they've encountered while trying to solve other problems such as this one. StackOverflow has particular rules about what is a valid question to ask, usually requiring that they be on topic, detailed, and not a duplicate of another question. There's a process for closing questions that violate these rules, however it requires manual intervention!

Let's say that hypothetically their product team has come to you (yes *you*) and wants to know: can they tell users that their post is likely to be closed for off-topic before they even post it?

Dataset

To save you some time, we've put together the dataset for you! It consists of 50k randomly sampled posts that have been closed as off-topic (indicated by 1 in the label columns) and 50k randomly sampled posts that haven't been closed and have an accepted answer (indicated by 0 in the label column).

In the provided CSV (download it here, note that it's about 215MB) you'll find five columns:

- Title
    - This is the title of the post that a user has submitted.
- Body
    - This is the body of the post that the user submitted.
- label
    - Either 0 or 1 to indicate acceptable versus off-topic posts respectively
- Title_processed
    - See below
- Body_processed
    - See below

To save you some time, we've also run the post Body and Title through a text normalization pipeline. These are in columns Body_processed and Title_processed respectively -- however

you are free to ignore them and write your own text processing pipeline if you think it's worthwhile!

Actual Problem

This problem is fundamentally a predictive analytics problem -- investigate and develop an algorithm that is able to classify whether or not a given post is going to be closed as off-topic, using solely the provided Body and Title. While we're not going to require that you develop a machine learning approach to this problem, we think it's probably appropriate -- and if you find that a heuristic works better, we'd want to see it analyzed the same way you would analyze an ML solution!

Submission

From you, we're expecting the following:

1) Code that ingests the data and produces a method for making the off-topic versus acceptable predictions.
    a) Please provide instructions for running the code.
    b) If there are any external dependencies needed to run the code, please be sure to specify them.
        i) For example, in Python with a requirements.txt
2) A report on the method you came up with, covering broadly:
    a) Why you chose the method that you chose.
    b) How effective the chosen method is at solving this problem.
    c) Some other questions detailed in the section below
3) Any analysis code that backs the report above.

Evaluation

We're not interested in the most effective method for this problem!

We're far more interested in you demonstrating your thought process for how you approached this problem, and an understanding of where your method's strengths and weaknesses lie.

There's some specific stuff we'd like to see discussed (to some degree) in that report, as detailed below:

1) What metric did you use to judge your approach's performance, and how did it perform? Why did you choose that metric?

2) The dataset we've given you is artificially balanced such that there's an even split of closed posts to accepted posts. Should this influence the metrics you measure?
3) How generalizable is your method? If you were given a different (disjoint) random sample of posts with the same labeling scheme, would you expect it to perform well? Why or why not? Do you have evidence for your reasoning?
4) How well would this method work on an entirely new close reason, e.g. duplicate or spam posts?
5) Are there edge cases that your method tends to do worse with? Better? E.g., How well does it handle really long posts or titles?
6) If you got to work on this again, what would you do differently (if anything)?
7) If you found any issues with the dataset, what are they?

Errata

1) While you're free to use whatever programming language you want, make sure you provide instructions for getting it to work. If you need help choosing, it would be easiest for us if you use Python.
2) Please provide any code that you used to answer the questions above.
3) We're expecting you to spend about two to three days on this. If you don't feel like that's enough time, feel free to ask for an extension.
4) Providing us the code in a Github repository is not required, but it is recommended.
5) The dataset is utf-8 encoded, so keep that in mind when trying to read it.