## Sentiment of Reviews - Project on Mongodb

Given a set of reviews, you will develop a super simple “sentiment” analyzer for these reviews.

From two collections of words (positive, negative respectively), you will have to count which words prevail in a review (algorithm below). You can also do it your own way. That’s fine too – as long as you can explain it. We have provided two lists of words for you, one is negative word list and the other is positive words list. Positive words are words like “superb”, “like”, “terrific”, “success” and etc. Negative words are words like “boring”, “annoy”, “hate” and etc. Feel free to find larger lists on internet and use them, there are plenty of such lists available.

Sample of reviews (which we have taken from kaggle.com) are provided and represented in JSON format (soon to be posted in sakai resources)

The task is to categorize reviews as positive or negative. You should give three examples of reviews with categories which are assigned by your program.

These three examples (your output) should be in JSON format, including the review category label (Positive or Negative)

Example

{ id: “5201\_1”, review: “I like the movie, it’s fantastic although I hate the actor James in the movie”, category: “positive”}

(two positive words and one negative. It’s a positive review based on the algorithm below)

The algorithm you can use is as follows.

Set sentiment score is 0;

Scan each word A in a review R

If A is a positive word, sentiment score = sentiment score+1;

If A is a negative word, sentiment score = sentiment score -1;

After scanning all words, if sentiment score ≥ 0, the review is positive; if the sentiment < 0, the sentiment score is negative.

Submit 5-10 power point slides (ready to present when asked). You should have a few slides on what in Mongo, what it is good for and how did you use it to deal with this problem.

You should also show SQL solution (to contrast with the NoSQL one), by providing a scheme for this problem (table names, not instances), and SQL query which would return the labels of individual reviews.

You should submit not only a result but also how you got the result and each procedure you have done including the runnable code.

**OPTION**: Feel free to illustrate NoSQL on your own project data, and use any NoSQL system you want. This will be rewarded with extra credit (if it is done well). Regardless which system you will use, you need to have a few slides explaining this system and of course what you did/

**Datasets Description:**

*UnlabelReview.json*: It contains review id and review contents, the review is raw data from costumers, organized in paragraph.

Example: { id: “5201\_1”, review: “I like the movie, it’s fantastic although I hate the actor James in the movie”}

*UnlabelReviewAfterSpliting.json*: It contains review id and review contents, each review has been split to word and word count. It is processed from UnlabelReview.json. We have omitted part of useless words, like stop words, like “is”,” are”, “do” and so on).

Example: { id: “5201\_1”, review: [{word: “like”,count:2}, {word: “movie”,count:2}, {word: “actor”: 1}, {word: “fantastic”, count:1}, {word: “hate”, count: 1} ]}

**Additional File:**

*positive words.txt*: a list of positive words you may use when categorizing the review

*negative words.txt*: a list of negative words you may use when categorizing the review