QnA Matching for Stackoverflow

Siddhant Sahu, Ram Anand Vutukuru

July 22, 2018

Problem Description This project will address the problem of determining if a question is a duplicate of some other question in the dataset. We consider a question duplicate of another question if a particular answer is a valid answer to both the questions.

Dataset Details There are four files we will work with:

1. Original questions dataset: It contains the list of all original questions – `questionId`, `creationDate`, `text`, `answerId`.
2. Duplicate questions dataset: It contains the list of question that are marked as duplicate of some other question (usually an original question) – *same schema as above.*
3. Answers dataset: It contains the list of all answers – `answerId`, `text`
4. Tags dataset: Each question will have several tags associated with it.

The raw files (i.e. the data dump) are in a different format (xml) and will be converted to the above format.

Approach The steps required are as follows:

1. Clean and process the data. This will involve stripping html tags, blocks of code and links from the question text.
2. Write a custom tokenizer to tokenize sentences to useful phrases, in addition to words. This has been known to improve performance of text classification. *Bonus feature: will include this if time permits.*
3. Extract text features such as TF-IDF and word2vec embeddings (*bonus*).
4. Train classifier(s) and evaluate models. *Bonus: Since this is can also be formulated as a ranking problem, we will explore this as a learning-to-rank problem if time permits.*

Data Selection & Evaluation Strategy

1. We will choose a few tags (common tags like `python`, `java`, etc that have a large number of duplicate questions) and train a classifier for each tag.
2. In addition, we will filter the dataset to keep those `answerId`s that are linked to at least 10-15 questions. This will ensure we have sufficient data per class, for training the classifier.
3. The classifier will use the `text` of question as feature and `answerId` as label (*recall how one answer can answer multiple equivalent questions*).
4. We will use *mean rank*, i.e. the mean position of the correct answer in the list of all answer classes, and *top 3 percentage*, i.e. the percentage of test questions that have the correct answer in the top 3.

Coding Language

1. Python scripts to convert data dump (raw xml files) to format described in **Dataset Details**. This is a one-time thing.
2. Spark project with Scala to build the text classifier.