**IST-736 Text Mining**

**Homework-5**

Training/Evaluation Data Acquisition Through AMT

Sharat Sripada (vssripad@syr.edu)

### Introduction

This week’s homework is around an interesting exploration about how a platform like Amazon Mechanical Turk (AMT) can be leveraged to label data. Subsequently, we will employ techniques like Cohen’s Kappa to determine the level of agreement or disagreement between the Turkers.

In the report we will briefly introduce the AMT platform, define important terminology, and see how it can be used to label the dataset introduced in Homework-1. That is, a corpus of tweets related to topic artificial intelligence.

### About the AMT platform

A common requirement with datasets in machine-learning or artificial intelligence is to have the data labelled sufficiently so models can be trained and eventually make predictions. For the dataset specifically, we are looking to tag sentiment on a bunch of tweets.

Using the AWS credentials, login to URL [www.mturk.com](http://www.mturk.com)

The platform offers role as *Worker or Requester*. To stage data and getting the AMT workforce to tag the data, we would need to use the role of *Requester* hence. Further, on sign-on select *New Project* and choose *Sentiment Analysis under Language* and hit the *Create Project* button as shown below:

Graphical user interface, text, application

Description automatically generated

Prior setting up the task it is important to understand the terminology. Following are a few:

* Worker: A worker refers to anyone with an MTurk worker account. Workers browse tasks posted on MTurk and choose to accept a task, work on it, and submit it when it is done.
* HIT: Expands as Human Intelligence Task. A HIT is a single unit of work that you want to complete. For example, we will use a collection of 30 tweets, each of those tweets would be a single HIT.
* Assignment: You can request one or more workers to complete each of your HITs. The work submitted by each worker for each HIT is called an Assignment. In our exercise, we will use 2 workers for 30 tweets amounting to a total of 60 assignments.

Knowing this helps us define the task clearly:

Graphical user interface, application, Teams

Description automatically generated

Next, we design the layout or dashboards the worker will see. Here, I customized the xml to comprise only Positive, Negative and Neutral sentiment (and removed the N/A). Also, we provide instructions to help the workers:

<full-instructions header="Sentiment Analysis Instructions">

<p><strong>Positive</strong> sentiment include: joy, excitement, delight</p>

<p><strong>Negative</strong> sentiment include: anger, sarcasm, anxiety</p>

<p><strong>Neutral</strong>: neither positive or negative, such as stating a fact</p>

Finally, the preview and Finish screen would like this:

Graphical user interface, application

Description automatically generated

### Data source, EDA, and labelled data on AMT

Once the dashboard and task has been setup on AMT, the final step is to Publish the data. For this purpose, we would create a csv file with the following content:

Graphical user interface, application, table

Description automatically generated

Where the column *text,* comprises tweets obtained using the *tweepy* library. However, prior writing tweets into the data-frame and csv file we filter, and clean tweets based on the following:

* Remove RT tags from each tweet using re.sub()
* If a http(s) URL appears in a tweet, we will pass since we do not expect Turkers to spend time researching on the URLs

#### Obtaining results from AMT platform

Having uploaded the data to AMT, we can now publish it and allow workers to work on tagging the sentiment on tweets. The task can be monitored in the Manage section and completed in a span of a few minutes. Below is a snapshot of the completed assignment and task:

Graphical user interface

Description automatically generated

Once the results are available in AMT, the data can be downloaded for further analysis (as csv file). The csv is appended with an additional column, comprising sentiment tags obtained from the NLTK sentiment intensity analyzer. Below is a snapshot of the resulting data-frame:

Graphical user interface

Description automatically generated with medium confidence

### Results and accuracy

In this section, we will use the Cohen’s Kappa score to correlate and compare the sentiment tagging between the Turkers and NLTK Sentiment Analyzer – in the following combinations:

* Between Mturk-1 and Mturk-2
* Between NLTK-Sentiment Analyzer and Mturk-1
* Between NLTK-Sentiment Analyzer and Mturk-2

But first, what is a Kappa score. Kappa score finds origins in the field of psychology, and it used for measuring the agreement between two human evaluators or raters (e.g., psychologists) when rating subjects (e.g., patients). It was later appropriated by machine-learning to measure classification performance. The metric is also commonly known as Cohen’s Kappa coefficient after the American statistician and psychologist Cohen Kappa.

Kappa can be commonly defined as:

KappaScore = (Agree - ChanceAgree) / (1 - ChanceAgree)

where:

* Agree - proportion of agreement between two parties
* ChanceAgree - expected proportion of chance agreements
* KappaScore - (-1, 1) where 1 is perfect agreement

#### Calculation of Kappa score using manual method

To understand the process thoroughly, see manual calculation of Kappa score between the two Turkers that is, Mturk-1 and Mturk-2.

Table

Description automatically generated

#### Calculation of Kappa score using sklearn

Using sklearn the Kappa score can also we calculated using Python as seen below:

Calendar

Description automatically generated with low confidence

**NOTE:**

The Kappa score between the manual method and the sklearn match. It is -0.05

Summary of results:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Between Turkers** | **Mturk-1 vs NLTK sentiment analysis** | **Mturk-2 vs NLTK sentiment analysis** |
| **Kappa score** | -0.05 | 0.04 | -0.14 |

### Conclusion

First, what a great platform - AMT! For first time users, it is simple, intuitive and offers just enough customizations to be in control of data. As for the Kappa score, we see the following:

* + Between Turkers - A negative score of 0.05 shows disagreement
  + Between Mturk-1 and NLTK sentiment analysis - A score of 0.04 shows mild agreement. Note that, since NLTK sentiment analysis was used in homework-1 to label sentiment, it was used as one of the evaluators
  + Between Mturk-2 and NLTK sentiment analysis - A negative score of 0.14 again shows disagreement.

Essentially showing poor agreement between any two columns in the data-frame.

Being curious if some of the Turkers were using a common NLTK sentiment tagger, comparisons between each Turker and the standard NLTK library were run. Again, very marginal agreement for one of the Turkers thus showing they possibly used discretion when sentiment tagging tweets.

Couple of possible improvements:

* In the mturk task there is possibility to request for more specific skills. I presume this would give us better or consistent results. This may be specific to the topic of artificial intelligence.
* Some of the tweets were truncated but I kept them assuming Turkers would tag it Neutral since I had a specific instruction, but they ended up tagging Positive/Negative for few instances. Perhaps, providing full tweets would aid Turkers better tag them.