# Ghulam Ishaq Khan Institute (GIKI)

## Assignment # 1

| | |
|---|---|
| **Subject:** Natural Language Processing | **Course Code: AI** - 361 - A - Spring - 23 |
| **Class:** BS AI, **Batch:** Fall – 2020 | **Submission Deadline:** 05/Mar/2023 - Sunday (11:59 - PM) |
| **Course Instructor:** M. Qasim Riaz - Lecturer - FCSE | **Total Marks:** 50 (Marks are divided problem wise) |
| **Course TA:** Mr. Ali Aftab | |

**Note (Read notes & instructions first)**

- First of all, read the instructions and statements of each exercise/question carefully then write the solution.
- It is written in front of each question that you have to upload it as handwritten or Python Notebook File.
- *For Handwritten:*
  - In case of multiple questions, give heading of each question's number or the exercise you are going to solve (don't write statement of question)
  - Mention page number and your roll number at corner of each page.
  - Take pictures using applications like camscanner on your phone.
  - Then select all pictures in the camscanner application and convert them into a pdf file using option in application.
  - The name of your pdf file should contain your assignment number and your roll number as shown in following example, For Example if your roll number is 2022532 and you have done assignment number 2 then the name of file should be as ---> 2022532_2.pdf
  - Then upload that pdf file at Microsoft teams. Remember the sequence of pages should be right.
  - Also keep the same original pages/hardcopy with you so that you can show/submit me later if required.
- *For Jupyter Notebook Code File:*
  - Create different file for each question & assignment
  - Name of each file should contain your roll number, assignment number & question number in a specific format.
  - For Example, if your roll number is 2022532 you are doing 2nd assignment and question no 5 then file name of your Python Notebook file should be written as ---> 2022532_2_5.py (Similarly, create for each question)
  - Now upload all of these files at Microsoft teams.

**CHEATING/COPY CASE or LATE SUBMISSION even 1 minute late will be graded as STRAIGHT ZERO MARKS.**

**So be on time make no excuse.**

## NLP Based Analysis of Twitter Samples

**Twitter Samples Data-Set:**

The Twitter data set in the NLTK (Natural Language Toolkit) corpus is a collection of 20,000 tweets that have been manually annotated with sentiment labels. The tweets were gathered using the Twitter API and cover a range of topics such as sports, politics, and entertainment.

The sentiment labels indicate whether the tweet is positive, negative, or neutral. The labels were assigned based on the presence of certain keywords and emoticons(emojis) that are commonly associated with positive or negative sentiment.

*To find the structure of stored tweets, labels and etc. in Twitter_Samples dataset, check these links:*

- *Link-1: NLTK Twitter Corpus*
- *Link-2: NLTK.org/Twitter-Samples*

**Your Tasks:**

In this assignment, you will be working with a dataset of tweets from Twitter. Your tasks are mentioned as following:

1. Load the dataset and perform exploratory data analysis (EDA) - also show Graphical Visualization:
   1.1. Count the number of positive, negative, and neutral tweets in the data set
   1.2. Analyze the frequency of words in the positive, negative and neutral tweets and print the 10 most common words in each category

1.3. Analyze the frequency of hashtags in the positive, negative and neutral tweets and print the 10 most common hashtags in each category

1.4. Analyze the frequency of retweets in the positive, negative and neutral tweets and print the 10 most common retweet counts in each category

2. Preprocess the text data by performing the following steps:

2.1. Tokenization:

Splitting the text into individual words or tokens.
*(Hint: Use Reg-Exp or NLTK – Tokenizer)*

2.2. Stopword Removal:

Removing commonly used words such as "the", "and", and "is" that do not add much value to the analysis.
*(Hint: Use Stopword corpus in NLTK - Helping Link: Stopwords Removal )*

2.3. Normalization:

Converting all text to lowercase to ensure consistency in the analysis.
*(Hint: Use Reg-Exp or String Manipulation Functions)*

2.4. Extraction of words starting with Hashtag (#) and Mention (@):

Identifying hashtags (#NLP) & mentions (i.e., @username) and convert to list and dictionary.
*(Hint: After tokenization and above steps use Reg-Exp to identify words starting with # sign or @ sign and store # sign words in list of hashtags and store @ sign words in username tags. Don't remove these words also keep them intact as normal token in tweets so that they can also be used later on. Have a look at following example 1.1 tweet and its tokens.)*

Example 1.1: Sample Tweet conversion to tokens with # and @.

I *@m_qasim* is excited to teach *#NLP* course *@GIKI* in *#2023*.
['@m_qasim', 'excited', 'teach', '#nlp', 'course', '@giki', '#2023']

Now above tweet is converted to token along with stop-words removal and text normalization. You can see all words with # and @ are still present that what I want. But you will have to store both # and @ words in separate lists for both just to keep records. You can also create a master list consisting of dictionaries that contains #words, @words and corresponding text. As shown below:

```
[{'hashtags': ['#nlp', '#2023'], 'usernames': ['@m_qasim', '@giki'], 'Text': ['@m_qasim', 'excited', 'teach', '#nlp', 'course', '@giki', '#2023']}, ...]
```

Similarly, this master list will contain dictionaries of all tweets with #, @ and text of each tweet.

2.5. URL Extraction:

Identifying URLs and replacing them with a common token named as 'url' every place where a url is found.
*(Hint: Use Reg-Exp to find the URL / Web Links out of each tweet's text and store them also in a list and add to above dictionary similarly as we did for @ sign and # sign words. Then remove the URL from original text of the tweet. And replace with word URL at the same point. Helping Link: Extract URLs)*

2.6. Removing emoticons / emojis:

Emoticons/ emojis are used very frequently at online platforms like twitters. We need clear them out from text in this phase.
*(Hint: Most of the emoticons are removed with stop words but to make sure we can use Reg-Exp to check if any them still exists in text and can be removed. For this purpose get help from this link. The following table shows the list of common emoticons. Helping Link: How to Remove Emoticons)*

**Examples-**

| | |
|---|---|
| :-) | Smile |
| ;-) | Smile with a wink |
| :<}) | User with moustache, smiling |
| :-\|\| | Mad |
| :-( | Sad |
| :'-( | Crying:~Also crying |
| :-)) | Really happy:-DBig grin |
| :-* | A kiss |
| :-P~ | A lick |
| :-o | Wow! or I'm surprised |
| :-\| | Grim:-PSticking out your tongue |
| :- | User happens to be Popeye |
| :-/ | Perplexed |
| =:O | Frightened (hair standing on end) |

3. Perform bag-of-words (BoW) representation on the preprocessed text data.

3.1. Use the BoW representation to identify the most frequent words in the positive, negative, and neutral tweets. Using CountVectorizor.

3.2. Show graphical visualization of top 10 in each category.

4. Perform part-of-speech (POS) tagging on the preprocessed text data.
   4.1. Identify the most common POS tags in the positive, negative, and neutral tweets at words level not in n-grams format.
   4.2. Show graphical visualization of top 10 in each category.
5. Use n-grams to identify common phrases in the positive, negative, and neutral tweets.
   5.1. Apply stemming or lemmatization to reduce words to their base form at pre-processed text.
   5.2. Create n-grams (unigrams, bigrams, and trigrams) from the preprocessed text data.
   5.3. Calculate the frequency of each n-gram (i.e., count how many times each n-gram appears in the text data)
   5.4. Sort the n-grams by frequency in descending order.
   5.5. Select the top 10 most frequent n-grams for each type of n-gram (i.e., unigrams, bigrams, and trigrams).
   5.6. Show graphical visualization of top-ten Uni-gram, Bi-gram, Tri-grams in all positive, negative and neutral tweets of data-set.

**Deliverables Your submission should include the following:**

- A Jupiter notebook with your code and analysis.
- Handwritten Report of containing flow (in form of Chart) of your work and explanation of different functions you used to complete each of the above-mentioned task.
  - For example, if you used NLTK, Reg-Exp, Matplotlib library to complete the above-mentioned task 1.4. Then in hand written document give heading of your task number and under this heading mention the libraries you used and specific functions of libraries you used, also mention the sample input you provided to each function and what it returned you back. Similarly do this for all mentioned tasks.
  - You don't need to write code just write name of operations you performed by showing sample input to output as we can show that in tokenization, we give text in string format and gets tokens in list format. Similarly, you have to show input and output of each operation you performed.

**Note: Any student proved/caught with any kind of cheating/plagiarism/late submission will be subject to zero marks**

**o --- | --- Good Luck --- | --- o**