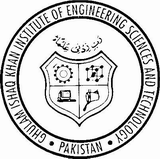
**Ghulam Ishaq Khan Institute of Engineering Sciences and Technology**

**Time: 120 minutes CS 251L FINAL Exam Spring 2023 Marks: 50**

****

|  |  |
| --- | --- |
| **VETTED BY** |  |

**Read the following instructions carefully.**

* **Program Structure:** Your program's structure should follow a specific order. Firstly, import modules. Secondly, define all functions. Thirdly, start with the main program. Make sure that non-indented lines are not imports or definitions.
* **Use PEP-8 Guidelines:** Use the PEP-8 guideline for the implementation of your code. This will make your code more readable, easier to maintain, and help you avoid common errors.
* **Proper Function Design:** Ensure that you have a proper function for everything. This includes loading and pre-processing data, plotting graphs, and creating models.
* **Comments and Code Clarity:** Make good use of comments. They can help clarify your code and ensure that other users can easily understand it. Additionally, use meaningful variable names, which can reduce the need for commenting. Think about future use cases when writing your code. (your code should have at least 6 commits.
* **Commit Code Frequently:** You should commit your code to GitHub every 15 minutes. This will ensure that you have a backup of your work and help you avoid losing any progress. Failure to do so may result in marks deduction.
* **Result Function:** You must have a function that takes any model from the code (e.g., naive bayes or KNN) and prints its result.
* **Save Plots:** Ensure that you save all plots that you create using "plt.savefig", your save fig should have 300 dpi.
* **Submission:** Your file name should be your regno.py, (2020XXX.py). You must submit a .py file. This will allow the instructor to run your code and evaluate it more efficiently.
* **Plagiarism:** In case of plagiarism both codes will be marked 0.
* **Reference:** See reference page at last for the references.
* **Uploading:** You must upload your work in teams.

# QUESTION 01:

## About Dataset:

This dataset contains a collection of tweets with the hashtag #chatgpt. The tweets were scraped from Twitter and cover a range of topics related to the ChatGPT language model. The dataset includes the following information for each tweet:

* Tweet text
* User information (username, user ID, location, etc.)
* Tweet timestamp
* Retweet and favorite count
* Hashtags used in the tweet.
* URLs

The dataset provides a glimpse into the online conversation surrounding the ChatGPT language model and can be used for various natural language processing and machine learning tasks, such as sentiment analysis, topic modelling, and more. It allows understanding the community, the level of interest, and the use of ChatGPT.

LINK: <https://tinyurl.com/3eu93hat>

Or visit Kaggle and search for “chatgpt twitter”.

## What needs to be done?

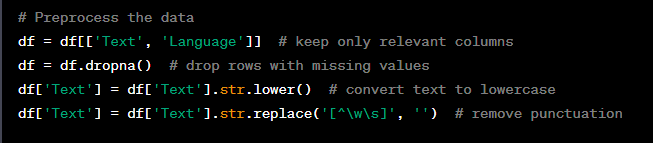
1. **Sentiment Analysis:** Can you predict the sentiment of tweets (positive, negative, or neutral) based on their text and other features such as hashtags and user mentions? You could use KNN, Naive Bayes, or Random Forest for this task, with sentiment labels as the target variable. (For labelling sentiments, you can use any library for adding/Labelling sentiments) See reference.
2. **User Classification:** Can you classify tweets based on the user who posted them? You could use KNN or Random Forest for this task, with user IDs(username) as the target variable.
3. **Clustering:** Can you cluster tweets into groups based on their text content or other features such as hashtags or user mentions? You could use K-Means clustering for this task, with tweet features as the input variables.
4. **Engagement Prediction:** Can you predict the number of retweets, likes, or replies that a tweet will receive, based on its text and other features? You could use KNN, Naive Bayes, or Random Forest for this task, with engagement metrics as the target variable.
5. **Hashtag Analysis:** Can you analyse the frequency and co-occurrence of hashtags in tweets, and identify common themes or topics? You could use K-Means clustering or Naive Bayes for this task, with hashtag counts as the input variables.

## Steps you can follow. (GUIDE)

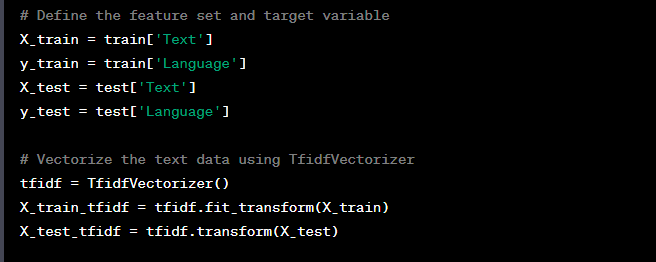
Here are some steps to apply the four supervised machine learning models and K-Means clustering to the given dataset:

1. Load the dataset into a panda DataFrame and perform any necessary data cleaning and pre-processing.
2. Split the dataset into a training set and a testing set.
3. Define the feature set and target variable. In this case, the 'Text' column can be used as the feature set and the 'Language' column can be used as the target variable.
4. Vectorize the text data using techniques such as Bag-of-Words or TF-IDF to convert it into a numerical format that can be used by the machine learning models.
5. Train each of the four supervised machine learning models (Decision Tree, Random Forest, KNN, and Multinomial Naive Bayes Classifier) on the training set and evaluate their performance on the testing set using metrics such as accuracy, precision, recall, and F1-score.
6. Apply K-Means clustering to the text data to cluster similar tweets together based on their text content. Choose an appropriate number of clusters and evaluate the performance of the clustering algorithm using metrics such as silhouette score and elbow method.

## Sample codes



A black screen with white text

Description automatically generated with low confidence 

A picture containing text, screenshot, font, line

Description automatically generated

# References

## Commit guide:

To Commit Your First Code:

1. Create a GitHub account and create a new repository for your code.
2. Initialize a git repository in your project folder by using the command: **git init**
3. Stage your changes by using the command: **git add .** (This will add all the files in your project folder to the staging area.)
4. Commit your changes by using the command: **git commit -m "First commit"** (This will create a commit with the message "First commit.")
5. Add a remote repository by using the command: **git remote add origin <repository URL>** (This will link your local repository to the remote repository you created in step 1.)
6. Push your changes to the remote repository by using the command: **git push -u origin master.**

**To Update Your Code via Commit:**

1. Make changes to your code in your local repository.
2. Stage your changes by using the command: **git add .** (This will add all the files you modified to the staging area.)
3. Commit your changes by using the command: **git commit -m "Message describing your changes"** (This will create a commit with a message describing the changes you made.)
4. Push your changes to the remote repository by using the command: **git push origin master.**

## PEP-8.

Follow the PEP-8 Make sure you do not loose marks for not adhering to them.

• One white space after commas, none before.  
• No space before and after binary operators (+, -, \*, /, \*\*) or one white space  
on both sides.  
• Always one white space before and after assignment = and comparison operators.  
The exception: no spaces around keyword argument equal signs.  
• Avoid lines longer than 79 characters. The vertical line in the spyder editor window  
shows the limit. Use continuation lines instead. (The 79 limit also applies to comments.  
• Always two empty lines before and after function definitions.

Using auto PEP-8 from the source pulldown menu in spyder will sort some PEP-8 problems, but not all.