1. Refer to the notebook on generating names using next-character prediction and modify it for generating text using **next-word prediction** (You have to implement MLP based text generator. However, it is recommended to refer to Andrej Karpathy's blog post on the Effectiveness of RNNs).

   Visualize the embeddings using t-SNE if using more than 2 dimensions or using a scatter plot if using 2 dimensions and write your observations. Write a streamlit application that asks users for an input text, and it then predicts the next **k** words or lines. In the streamlit app, you should have controls for modifying context length, embedding dimension, activation function, random seed, etc. You can use any one of the datasets mentioned below.

   Hints:

   a. For text-based datasets, you can remove special characters except "full stop (.)" so that it can be used to split sentences. However, you cannot ignore special characters for other datasets like for C++ code. You will have to treat text between newlines as a statement. To remove special characters from a line, you can use the following code snippet:

      import re
      line = re.sub('[^a-zA-Z0-9 \.]', '', line)

      It will remove everything except alphanumeric characters, space and full-stop.

   b. Convert the text to lowercase and use unique words to create the vocabulary.
   c. To create X, and y pairs for training, you can use a similar approach used for next-character prediction. For example:

```
. . . . . ---> to
. . . . to ---> sherlock
. . . to sherlock ---> holmes
. . to sherlock holmes ---> she
. to sherlock holmes she ---> is
to sherlock holmes she is ---> always
sherlock holmes she is always ---> the
holmes she is always the ---> woman
she is always the woman ---> .
```

      You will get something like *". . . . . ---> to"* whenever there is a paragraph change.
   d. You may have to use a larger embedding size for words. (For example: 32 or 64)
   e. Use a similar model as used for next-character prediction. Here, you may have to increase the size of hidden layers. (For example, 1024).
   f. For the streamlit app, no need to re-train the model based on the user input. Train two to three variants and accordingly give options to the user.

g. For visualizations, you may have to select words with relations like synonyms, antonyms, names and pronouns, verb and adverbs, words with no relations, and so on.
h. Think how you would handle the case where words provided by the user in streamlit are not in the vocabulary.
i. Use Google Colab or Kaggle for training (use maximum 500-1000 epochs). Start the assignment early, as training takes time.

Datasets:

a. Paul Graham essays
b. Wikipedia (English)
c. Shakespeare
d. Leo Tolstoy's War and Peace
e. The Adventures of Sherlock Holmes, by Arthur Conan Doyle
f. Maths texbook
g. Python or C++ code (Linux Kernel Code)
h. IITGN advisory generation
i. IITGN website generation
j. Generate sklearn docs
k. Notes generation
l. Image generation (ascii art, 0-255)
m. Music Generation
n. Something comparable in spirit but of your choice (do confirm with TA Anupam)

[5 marks]

2. Learn the following models on XOR dataset (refer to Tensorflow Playground and generate the dataset on your own containing 200 training instances and 200 test instances) such that all these models achieve similar results (good). The definition of good is left subjective – but you would expect the classifier to capture the shape of the XOR function.
    a. a MLP
    b. MLP w/ L1 regularization (you may vary the penalty coefficient by choose the best one using a validation dataset)
    c. MLP w/ L2 regularization (you may vary the penalty coefficient by choose the best one using a validation dataset)
    d. learn logistic regression models on the same data with additional features (such as x1*x2, x1^2, etc.)

    Show the decision surface and comment on the plots obtained for different models. [2 marks]
3. Train on MNIST dataset using an MLP. The original training dataset contains 60,000 images and test contains 10,000 images. If you are short on compute, use a stratified subset of a smaller number of images. But, the test set remains the same 10,000 images. Compare against RF and Logistic Regression models. The metrics can be:

F1-score, confusion matrix. What do you observe? What all digits are commonly confused?

Let us assume your MLP has 30 neurons in first layer, 20 in second layer and then 10 finally for the output layer (corresponding to 10 classes). On the trained MLP, plot the t-SNE for the output from the layer containing 20 neurons for the 10 digits. Contrast this with the t-SNE for the same layer but for an untrained model. What do you conclude?

Now, use the trained MLP to predict on the Fashion-MNIST dataset. What do you observe? How do the embeddings (t-SNE viz for the second layer compare for MNIST and Fashion-MNIST images) [3 marks]

Submission Format: Share a GitHub repo with your training notebooks named *"question<number>.ipynb"*. Include textual answers in the notebook itself. For Question 1, put the link to streamlit app at the top of the notebook.