# ML Learners' Space

**Assignment - 2**

Hello, we hope you enjoyed learning this week and gained valuable insights. Let's apply your knowledge with the following problems!

## Problem 1

Use the popular SMS Spam Collection dataset (available on Kaggle), which contains labeled messages as either "spam" or "ham" (not spam), stored in a Pandas DataFrame with columns Label (spam/ham) and Message (text). Perform the following tasks:

- Preprocess each message by tokenizing, removing stop words, and lowercasing the text.
- Load the pre-trained Google News Word2Vec model using `gensim`.
- Convert each message into a fixed-length vector by averaging the Word2Vec vectors of all the words in the message (ignore words not found in the model vocabulary).
- Split the dataset into training (80%) and testing (20%) sets using `train_test_split`.
- Train a Logistic Regression classifier on the vectorized training data and print the accuracy on the test set.
- Write a Python function `predict_message_class(model, w2v_model, message)` that takes a trained classifier, the Word2Vec model, and a single message (string), and returns the predicted class (spam or ham).

## Problem 2

Use the Twitter US Airline Sentiment dataset (available on Kaggle), which contains tweets labeled with the sentiment of the user toward airlines (positive, negative, or neutral). The data is stored in a Pandas DataFrame with columns such as `airline_sentiment` (target) and `text` (tweet content). Perform the following tasks:

- Preprocess each tweet using the following steps:
  - Convert the text to lowercase.
  - Remove URLs, mentions (e.g., `@username`), hashtags, and punctuation.
  - Expand common contractions (e.g., `"don't"` → `"do not"`).
  - Lemmatize the words (use NLTK).
  - Optionally remove emojis and special symbols.
- Load the pre-trained Google News Word2Vec model using `gensim`.

- Convert each tweet into a fixed-length vector by averaging the Word2Vec word vectors for all words in the tweet. Ignore words not found in the embeddings.
- Split the dataset into training (80%) and testing (20%) sets using `train_test_split`.
- Train a Logistic Regression classifier on the vectorized training data and report the accuracy on the test set.
- Write a Python function `predict_tweet_sentiment(model, glove_model, tweet)` that takes the trained classifier, the GloVe model, and a single tweet (string), and returns the predicted sentiment (positive, negative, or neutral).