**Team Name : Astra**

**Group Members :**

Kartik Shetty
Manthan Dhole
Kashyap Chavhan
Hrishikesh Bade

**Approach Method:**

This is a Python code for a machine learning model that uses linear regression to predict the length of a product based on its tags. Here is an overview of the code:

- ➢ The code imports necessary libraries such as pandas, numpy, re, nltk, and sklearn.
- ➢ The code loads a dataset from a CSV file using pandas' read_csv function.
- ➢ The code drops any rows with missing values in the "TITLE" column using the dropna function.
- ➢ The code fills any remaining missing values with empty strings using the fillna function.
- ➢ The code combines the "TITLE", "BULLET_POINTS", and "DESCRIPTION" columns into a single "TAGS" column.
- ➢ The code preprocesses the text in the "TAGS" column by converting it to lowercase, removing punctuation and digits, removing stop words and "dirt" words (a custom list of characters), and stemming the remaining words.
- ➢ The code drops unnecessary columns from the dataset.
- ➢ The code splits the dataset into training and testing sets using train_test_split function from sklearn.
- ➢ The code converts the text in the "TAGS" column to a matrix of word counts using CountVectorizer function from sklearn.
- ➢ The code trains a linear regression model on the training set using the LinearRegression function from sklearn.
- ➢ The code evaluates the model's performance on the training set and testing set using mean squared error metric.
- ➢ Finally, the code prints the training and testing mean squared error.
- ➢ In summary, this code preprocesses text data, converts text to numerical vectors, trains a linear regression model, and evaluates its performance using mean squared error.

Also we applied various other method like neural network:

- This is a Python script that demonstrates how to train a neural network model using Keras to predict the length of a product based on various features, and then use the trained model to generate predictions on a test dataset.

- The script begins by importing necessary libraries including numpy, pandas, scikit-learn, and Keras. It then loads the training data from a CSV file using Pandas, and uses the train_test_split function from scikit-learn to split the data into training and validation sets. The "PRODUCT_ID" and "PRODUCT_LENGTH" columns are dropped from the data, as the former is not needed for training and the latter is the target variable.

- Next, a StandardScaler object from scikit-learn is used to standardize the numerical features in the training and validation sets. The script then defines a Keras Sequential model with three layers: two hidden layers with 64 and 32 neurons respectively, both using ReLU activation functions, and an output layer with a single neuron (since this is a regression problem). The model is compiled with the mean_absolute_percentage_error loss function and the Adam optimizer.

- The model is then trained on the standardized training data using the fit method, with the validation data passed as an argument to track model performance on the validation set during training. The script trains the model for 10 epochs, with a batch size of 64.

- Finally, the script loads the test data from a CSV file and standardizes the numerical features using the same StandardScaler object used on the training data. The trained model is then used to predict the "PRODUCT_LENGTH" column of the test data, and the predictions are stored in a Pandas DataFrame along with the corresponding "PRODUCT_ID" values. The DataFrame is saved to a CSV file for submission.