Bank Marketing (Campaign)3

- **Team member's details :**

  Team Me, Francisco Lopez, panch.lopez.21.1998@icloud.com, United States, Eastern University, Data Science

- **Problem description:**

  ABC Bank wants to sell its term deposit product to customers and before launching the product they want to develop a model which helps them in understanding whether a particular customer will buy their product or not (based on customer's past interaction with bank or other Financial Institution).

- **Github Repo link:**https://github.com/Panch2/Bank-Marketing-Campaign-.git

- **Data cleansing and transformation done on the data:**

1. **Handling Missing Values:** You filled missing values in the 'job' column with the mode and replaced missing values in the 'education' column with 'unknown'.
2. **Data Transformation:** You converted categorical variables into binary indicators using one-hot encoding for columns like 'marital', 'default', 'housing', 'loan', and 'month'.

- **Try at least 2 techniques to clean the data (for NA values : mean/median/mode/Model based approach to handle NA value/WOE and like this try different techniques to identify and handle outliers as well):**

  1. Handling Missing Values:

  a. Used the SimpleImputer class from scikit-learn to replace missing values with the mean of each numerical column.
  b. Applied the fit_transform method of SimpleImputer to replace missing values in the numerical columns of the DataFrame with their respective means.

  2. Handling Categorical Variables:

c. Encoded categorical variables using one-hot encoding (specifically, pd.get_dummies) to convert them into numerical format for machine learning models.

3.Model-Based Imputation (Optional):

d. Fitted a K-Nearest Neighbors (KNN) model to predict missing values in the 'job' column based on other features in the dataset.
e. Used the trained KNN model to predict missing values for the 'job' column in the test set.

- **For NLP try different featurization technique and also clean the data using regex and python:**

- Data Loading: You loaded a dataset using fetch_ucirepo from the ucimlrepo library. This dataset is related to bank marketing.

- Text Cleaning: You cleaned the text data using regular expressions (re module) to remove unwanted characters, symbols, and patterns.

- Tokenization: You tokenized the cleaned text, which involves splitting it into individual words or tokens. You used the word_tokenize function from the nltk.tokenize module for this purpose.

- Stopword Removal: Stopwords are common words (e.g., "the", "is", "and") that often don't contribute much to the meaning of a text. You removed stopwords from the tokenized text using the stopwords corpus from the nltk.corpus module.

- Lemmatization: Lemmatization involves reducing words to their base or root form. You performed lemmatization using WordNet's built-in lemmatizer from the nltk.stem module.

Bank Marketing (Campaign)3

- Vectorization: You converted the preprocessed text data into numerical representations suitable for machine learning models. Specifically, you used two techniques:
    a. Bag-of-Words (BoW): This technique represents text data as a matrix where each row corresponds to a document (or text sample) and each column corresponds to a unique word in the corpus. The values in the matrix represent word frequencies (count) in each document. You used the CountVectorizer from the sklearn.feature_extraction.text module for this.

    b. Term Frequency-Inverse Document Frequency (TF-IDF): This technique also represents text data as a matrix, but it takes into account the importance of words by considering their frequency in the current document (term frequency) and across all documents (inverse document frequency). You used the TfidfVectorizer from the sklearn.feature_extraction.text module for this.

- Feature Names: Finally, you printed the feature names (i.e., the unique words or tokens) for both the BoW and TF-IDF representations using the get_feature_names method of the vectorizers.