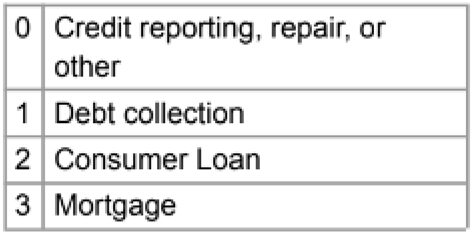
Task 6. Data Science example.

Perform a Text Classification on consumer complaint dataset

(https://catalog.data.gov/dataset/consumer-complaint-database) into following categories.



0	Credit reporting, repair, or other
1	Debt collection
2	Consumer Loan
3	Mortgage

Steps to be followed -

1. Explanatory Data Analysis and Feature Engineering

2. Text Pre-Processing

3. Selection of Multi Classification model

4. Comparison of model performance

5. Model Evaluation

6. Prediction

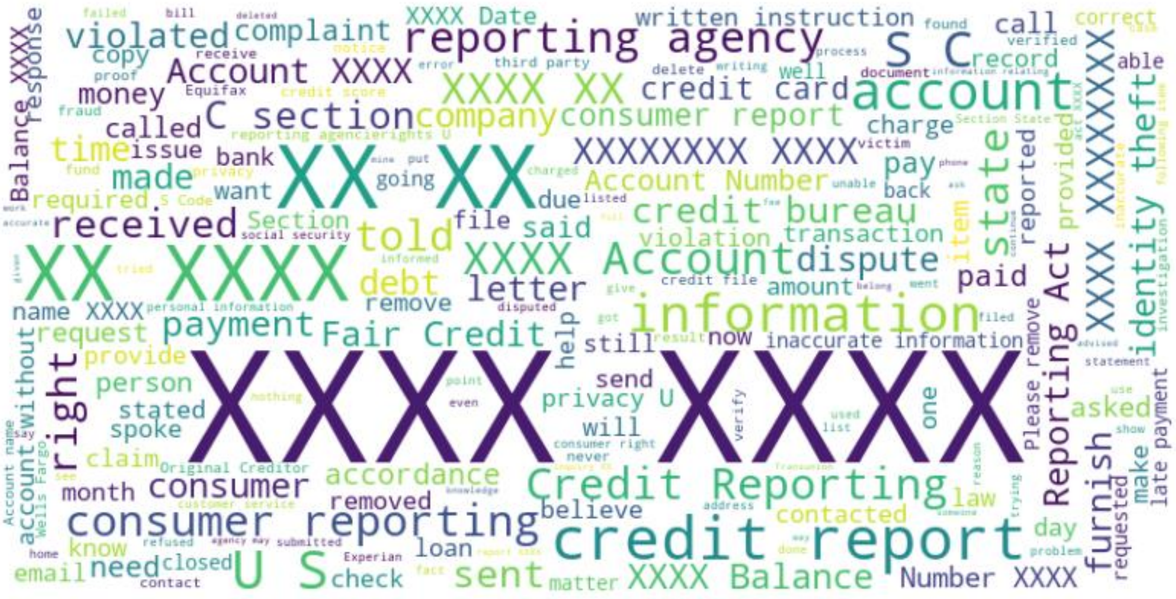
1) Explanatory Data Analysis (EDA):

Explanatory Data Analysis (EDA) is an essential initial step in data analysis, where the primary goal is to gain a deep understanding of the dataset you are working with. EDA helps data scientists and analysts to:

1.1) **Understand the Data:** EDA helps you familiarize yourself with the dataset, including its size, structure, and basic statistics. It allows you to identify the data types, missing values, and potential issues.

1.2) **Visualize Data:** EDA involves creating various visualizations, such as histograms, scatter plots, bar charts, and box plots, to visualize the distribution of data, relationships between variables, and potential patterns or outliers.

1.3) **Word Cloud:** In the context of text data analysis, a "Word Cloud" is a popular visualization technique that provides a visual representation of the most frequently occurring words in a corpus of text.



2) Feature Engineering:

Feature engineering is the process of creating new features (variables) or modifying existing ones to improve the performance of machine learning models. It involves a deep understanding of the data and domain knowledge. Key aspects of feature engineering include:

2.1)Feature Creation: Creating new features based on existing ones to capture relevant information. For example, creating a "total income" feature by summing up individual income components.

2.2) Encoding Categorical Variables: Converting categorical variables into numerical format, often using techniques like one-hot encoding or label encoding.

3) Text Pre-Processing:

Text Pre-Processing is a critical step in natural language processing (NLP) and text analysis. It involves cleaning and transforming raw textual data into a format that is suitable for further analysis, including machine learning and data mining. The primary objectives of text pre-processing are to enhance the quality of the data, reduce noise, and prepare it for feature extraction and modeling. Let's break down the key components of text pre-processing based on the provided code snippet:

3.1) Lowercasing:

In NLP, text data is often converted to lowercase to ensure uniformity. This step helps in treating words with different letter cases (e.g., "Word" and "word") as the same word. Lowercasing is beneficial for tasks like text classification and sentiment analysis.

3.2) Punctuation Removal:

Punctuation marks (e.g., commas, periods, exclamation marks) are typically removed from text because they often do not carry meaningful information for many NLP tasks. Removing punctuation simplifies the text and reduces dimensionality.

3.3) Tokenization:

Tokenization is the process of breaking a text into individual words or tokens. Tokens are the fundamental units of text that are used for analysis. Tokenization allows you to separate text into meaningful components, making it easier to work with and analyze.

3.4) Stopword Removal:

Stopwords are common words (e.g., "the," "and," "is") that do not carry significant meaning on their own and are often removed from text. Removing stopwords helps reduce noise in the data and focuses on content words.

3.5) Stemming (or Lemmatization):

Stemming is the process of reducing words to their root or base form (stem). This helps in treating variations of a word as the same word. Another option is lemmatization, which reduces words to their base form (lemma) using vocabulary and morphological analysis. Stemming and lemmatization can improve the efficiency and effectiveness of text analysis.

4) Selection of Multi-Classification Model:

When dealing with multi-class classification problems, where the goal is to assign an input data point to one of several predefined classes or categories, choosing the right classification model is crucial for achieving accurate and reliable results. Below are various models that are commonly used in multi-class classification tasks, along with brief explanations of each:

4.1) Decision Tree:

Theory: Decision trees are hierarchical structures that recursively split the dataset based on feature attributes. At each node, the decision tree algorithm selects the feature that best separates the data into different classes. Decision trees are interpretable and can handle both categorical and numerical data.

4.2) K-Nearest Neighbors (KNN):

Theory: K-Nearest Neighbors is a non-parametric classification algorithm that assigns a data point to the majority class among its k-nearest neighbors. It measures similarity based on distance metrics (e.g., Euclidean distance) and can be used for both binary and multi-class classification problems.

4.3) Naive Bayes (NB):

Theory: Naive Bayes is a probabilistic classification algorithm based on Bayes' theorem with the "naive" assumption of feature independence. It calculates the probability of a data point belonging to a particular class given its feature values. Naive Bayes is simple, computationally efficient, and works well with text and categorical data.

4.4) Stochastic Gradient Descent (SGD):

Theory: Stochastic Gradient Descent is an optimization algorithm that can be used with various classification models (e.g., linear classifiers). In the context of multi-class classification, it minimizes a loss function to find optimal model parameters. SGD is efficient and can handle large datasets.

4.5) Random Forest (RF):

Random Forest is an ensemble learning method that combines multiple decision trees to improve classification accuracy and reduce overfitting. It works by aggregating the predictions of individual decision trees. Random Forest is robust, handles high-dimensional data well, and provides feature importance scores.

4.6) Ensemble Voting Classifier:

An ensemble voting classifier combines the predictions of multiple individual classifiers to make a final decision. In this case, it consists of Decision Tree (DT), K-Nearest Neighbors (KNN), Naive Bayes (NB), Stochastic Gradient Descent (SGD), and Random Forest (RF). By combining the strengths of different algorithms, ensemble classifiers can often achieve higher accuracy and better generalization.

5) Comparison of Model Performance:

When working on a machine learning or data classification problem, assessing the performance of different models is essential to make informed decisions about which model to deploy for real-world applications. In this section, we'll compare the performance of various classification models based on their accuracy scores. Accuracy is a common evaluation metric that measures the proportion of correctly classified instances out of the total instances.

Here are the accuracy scores for different models:

- **Decision Tree (DT) Accuracy: 85%**
  Decision trees are interpretable and suitable for tasks where interpretability is important. An accuracy of 85% indicates that the decision tree correctly predicted the class labels for 85% of the instances in the test dataset.
- **K-Nearest Neighbors (KNN) Accuracy: 80%**
  K-Nearest Neighbors is a simple and intuitive algorithm. An accuracy of 80% means that KNN correctly classified 80% of the test instances based on their nearest neighbors in the feature space.
- **Naive Bayes (NB) Accuracy: 77%**
  Naive Bayes is known for its simplicity and efficiency. An accuracy of 77% indicates that Naive Bayes correctly predicted the class labels for 77% of the test instances, considering the independence assumption of features.
- **Stochastic Gradient Descent (SGD) Accuracy: 87%**
  Stochastic Gradient Descent is an optimization algorithm commonly used with linear classifiers. An accuracy of 87% suggests that SGD achieved a high level of accuracy in classifying instances.

- **Random Forest (RF) Accuracy: 86%**
  Random Forest is an ensemble method that combines multiple decision trees for improved performance. An accuracy of 86% indicates that the ensemble of decision trees provided accurate predictions.
- **Ensemble Voting Classifier Accuracy: 87%**
  The ensemble voting classifier combines predictions from multiple classifiers, including Decision Tree, K-Nearest Neighbors, Naive Bayes, Stochastic Gradient Descent, and Random Forest. An accuracy of 87% suggests that the ensemble achieved high accuracy by leveraging the strengths of various models

6) Model Evaluation:

In the context of machine learning, model evaluation is a critical step that involves assessing the performance and effectiveness of a trained model on unseen or real-world data. In this section, we'll discuss the process of model evaluation, specifically focusing on the chosen Stochastic Gradient Descent (SGD) model, which demonstrated both higher accuracy and efficiency.

- **Model Selection:**
  Before diving into model evaluation, it's important to briefly revisit the model selection process. In your case, the SGD model was selected based on its higher accuracy and faster training time compared to other models. The accuracy of 87% on a test dataset suggests that the SGD model performed well in classifying instances, and its computational efficiency makes it a suitable choice for real-world applications.

- **Running the SGD Model on the Whole Dataset:**
  After selecting the SGD model, the next step was to assess its performance on the entire dataset, rather than just the test set. Running the model on the entire dataset is a common practice to ensure that the model generalizes well to all available data. The accuracy of 87% on the whole dataset indicates that the model's performance remained consistent when applied to a larger and more representative sample.