**Project On Twitter Sentiment Analysis**

**Aim:-**To analyze the sentiment of user tweets regarding their opinions about the company, categorizing them as Positive, Negative, Neutral, or Irrelevant.

**Project Details**:-

1)**Dataset Overview:**

* The dataset was loaded from Kaggle and contains **4 columns**: Id, Companies, Label, and Text (Tweets), with a total of **74,863 rows**.

2) **Initial Data Transformation:**

* All the main libraries and functions required for the project were imported in the initial setup, focusing on **data visualization**, **text analysis**, **text vectorization**, and **model building**.
* The training and validation datasets were loaded into separate variables using the read\_csv function from **pandas**. These datasets did not include headers.
* Column names were renamed to match the data: Id, Companies, Label, and Text (tweets).

3)**Text Preprocessing:**

* A new column was created by converting all text data to lowercase using the str.lower() method.
* Some tweets contained only numerical values (e.g., a tweet with just the number "2"). To handle this, all data was transformed into strings.
* Special characters were removed using a **regex expression**, as Twitter data often contains typos or special characters.

4) **Data Visualization:**

* To identify the most frequently used words for each label, a **Word Cloud** was generated.
  + For instance, words like *love* and *game* were predominantly associated with positive labels, representing good sentiments.
* A bar plot revealed that games such as **MaddenNFL** and **NBA2K** had the highest number of negative tweets, while other brands displayed different trends.

5)**Token Analysis:**

* After cleaning the text, the total number of unique tokens was calculated, revealing more than **30,000 unique words**.
* A tokens\_text variable grouped all words from the text into a list.
* The main English **stopwords** were stored in a separate variable for use in later modelling steps.

6)**Logistic Regression Model:**

* A **Logistic Regression** model was built using the sklearn library, combined with the **Bag of Words (BoW)** approach for text vectorization.
* The BoW approach classified and grouped relevant data, enabling the model to identify trends effectively.
* The BoW model incorporated stopwords and used a default n-gram value of 1.

7)**Train-Test Split:**

* The dataset was split into **training** and **testing** subsets, with word encoding performed based on the training dataset as a reference.

**Insights:- Data Insights:**

* Upon analyzing the first 5 rows of the dataset, it was observed that a positive sentiment was assigned to a "kill" thread related to a videogame. Despite this anomaly, the modeling process adhered to a standard NLP project workflow.
* For the validation data, the first 5 rows did not exhibit any unusual labeling issues.
* The total number of tweets per category indicated that **negative** and **positive** sentiments were the most frequently recorded, while **irrelevant** tweets were the least common.
* Using this data, the Logistic Regression model was trained, achieving an **accuracy of 90%** on the test dataset and **98%** on the validation dataset.
* Subsequently, another Bag of Words model was applied, using a **4-gram approach** without classifying stopwords and leveraging all available information.
* The test dataset accuracy remained at **90%**, while the validation dataset accuracy reached **98%**, indicating that this approach outperformed the simpler n-gram and stopwords-based model.
* At first glance, the default parameters of the XGBoost model yielded a lower accuracy compared to other methods. Consequently, an additional step was added to analyze the model's training performance.
* It was determined that the XGBoost model had the lowest accuracy among all approaches, highlighting the need for parameter tuning to enhance performance.
* As demonstrated in the notebook, a straightforward NLP approach achieved **90% accuracy** on the test dataset, even with the labeling inconsistencies observed in the initial analysis.
* **Model Accuracy:** The model achieved an **accuracy of 90.81%**, indicating the proportion of correctly classified tweets across all categories.

**Performance Metrics per Label**

1. **Irrelevant:**

* **Precision:** Indicates that 96% of the tweets classified as irrelevant were actually irrelevant.
* **Recall:** Demonstrates that 85% of all irrelevant tweets were correctly identified by the model.
* **F1-Score:** 0.90% Balances precision and recall to provide a single performance measure.

1. **Negative:**

* **Precision:** Shows that 92% of the tweets predicted as negative were truly negative.
* **Recall:** Confirms that 92% of all negative tweets were correctly captured.
* **F1-Score:** 0.92% A strong balance between precision and recall for this label.

1. **Neutral:**

* **Precision:** Indicates that 91% of tweets classified as neutral were accurate.
* **Recall:** Highlights that 90% of all neutral tweets were identified correctly.
* **F1-Score:** 0.91% A reliable metric showcasing the model's performance for neutral sentiments.

1. **Positive:**

* **Precision:** 87% of tweets labeled as positive were actually positive.
* **Recall:** The model successfully captured 93% of all positive tweets.
* **F1-Score:** Demonstrates a good balance between precision and recall.