CRISP-DM DOCUMENT: TWITTER SENTIMENT ANALYSIS

1. Business Understanding

Goal:
This project aims to build a sentiment classification model that can detect the emotional tone of tweets related to Apple and Google products. The model will categorize each tweet as either Positive, Negative or Neutral based on the tweet text.

1. To analyze sentiment distribution, brand mentions, and class imbalance, by performing an exploratory data analysis to uncover key patterns in the Twitter dataset.

2.Sentiment & Brand Analysis:
The goal is to classify tweets about technology brands (Apple, Google, or none) into sentiment categories (positive, negative, neutral) to understand public perception and compare emotional engagement across brands.

3.Model Development:

- Binary classification modeling: To build, tune, and evaluate various machine learning models (Logistic Regression, Decision Trees, Random Forest to classify tweets as either positive or negative
- Multiclass classification modeling: To group the remaining emotion categories as 'neutral' then train models to further classify tweets as neutral

2. Data Understanding

The dataset comes from CrowdFlower and is available through Data.world. It includes 9,093 tweets manually labeled for sentiment. These tweets discuss Apple and Google, particularly around the 2011 SXSW tech conference. The columns in the dataset include:

- **tweet_text**: the text of the tweet itself

- **emotion_in_tweet_is_directed_at**: the specific brand or product being talked about
- **Is_there_an_emotion_directed_at_a_brand_or_product**: whether an emotion is clearly aimed at a brand/product

3. Data Preparation

We used an object-oriented approach to structure the data preparation steps in the notebook. The Explore class we created was responsible for inspecting the dataset by checking its shape, features, summary statistics, missing values, and duplicate rows. The Clean class handled the following:

- Removed duplicates to avoid repetition in training
- Handled missing values by:
    - Dropping a record with a missing tweet_text.
    - Replacing rows in the emotion_in_tweet_is_directed_at column with missing values with 'No Brand'

For text preprocessing, a dedicated class performed the following :

Preprocessing steps with NLTK:

- Lowercasing. Ensures 'Apple' and 'apple' are treated the same.
- Tokenization to break tweets into individual words.
- Stopword Removal to eliminate filler words like 'is', 'the', 'and'.
- Lemmatization to reduce words to their base forms which improves consistency in analysis

Feature Engineering:

We enriched the dataset by generating both linguistic and structural features from the tweet text. Specifically, we created the following:

- Character Count, Word Count, and Sentence Count – to capture the structure and length of each tweet.
- Cleaned Text – where unwanted elements such as URLs, hashtags, emojis, punctuation, numbers, and extra spaces were removed.

- Tokenized Text – where the cleaned text was broken down into individual words with stopwords removed.
- Lemmatized Text – where each token was reduced to its root form to normalize word variations.
- Document Column – created by joining the lemmatized tokens back into a single string, ready for vectorization.

This modular design streamlined the data preparation process, ensuring consistent and reusable methods for cleaning and preparing the data before analysis and modeling.

1. Modeling

Tools Used:

- NLTK: For cleaning and preparing the text
- Pandas: For data wrangling and structuring
- Scikit-learn: For building ML models
- XGBoost: For boosting performance using advanced tree- based methods

Two modeling approaches were used:

1. Binary Classification:

Goal: To predict if a tweet is Positive or Negative

Algorithms Used:

- Logistic Regression
- Decision Trees
- Random Forest
- XGBoost

2. Multiclass Classification:

Goal: To predict if a tweet is Positive, Negative or Neutral.

Algorithms Used:

- Multinomial Naive Bayes
- XGBoost

Model training involved splitting data into training and testing sets, training models, and evaluating them using cross-validation.

5. Evaluation

We evaluated all models using the following evaluation metrics:

- Accuracy: Overall Correctness.
- Precision: Accuracy of Positive Predictions.
- Recall: Ability to identify actual Positives.
- F1-Score: Balance between Precision and Recall.
- Confusion Matrix: Breakdown of Prediction Types.

Binary Classification Results:

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.85	0.84	0.86	0.85
Decision Tree	0.82	0.81	0.83	0.82
Random Forest	0.87	0.86	0.88	0.87
XGBoost	**0.88**	**0.87**	**0.89**	**0.88**

Multiclass Classification Results:

Model	Accuracy	Precision	Recall	F1-Score
Multinomial Naive Bayes	0.78	0.76	0.77	0.76
XGBoost	**0.80**	**0.79**	**0.81**	**0.80**

In this Twitter sentiment analysis project, accuracy was chosen as the primary evaluation metric because it provides a straightforward measure of overall model performance by indicating the proportion of correctly classified tweets out of all predictions. For an initial proof of concept, accuracy helps quickly assess whether the model is learning useful patterns before moving to more complex metrics.

Moreover, accuracy is widely understood and allows easy comparison between different models during early development. While it may not fully capture performance on minority classes in an imbalanced dataset, it offers a good starting point. To address imbalance concerns, other metrics like precision, recall, and F1-score can be used later to evaluate how well the model performs specifically on underrepresented classes.

6. Challenges

- **Class Imbalance**: The dataset had an imbalance in sentiment distribution, with more positive tweets than negative or neutral ones. This affected model performance, particularly in the multiclass classification.
- **Overfitting**: Some models, especially Random Forest and XGBoost, showed signs of overfitting, where they performed well on training data but poorly on unseen test data.
- **Complexity of Language**: Tweets often contained informal language, abbreviations, and emojis, which made sentiment classification challenging. The models struggled to capture the nuances of language used in tweets.

7. Conclusion and Recommendations

The sentiment analysis project has successfully demonstrated the potential of machine learning models to classify tweets related to Apple and Google products into positive, negative, and neutral sentiments. The binary classification models, particularly Logistic Regression and Decision Trees, provided a solid foundation for understanding sentiment distribution.

However, challenges such as class imbalance and overfitting were evident, especially in the more complex models like Random Forest and XGBoost. The multiclass classification models, including Multinomial Naive Bayes and XGBoost, showed promise in handling multiple sentiment classes but also faced similar challenges. The use of techniques like SMOTE for oversampling did not yield significant improvements, indicating that the dataset's quality and complexity may be limiting factors.

Next Steps:

- **Data Augmentation**: Explore additional data augmentation techniques to improve model robustness, especially for the minority classes.
- **Advanced Models**: Consider using more advanced models like BERT or other transformer-based architectures that can better capture the nuances of language in tweets.
- **Hyperparameter Tuning**: Further hyperparameter tuning of the existing models to optimize performance. In this project we were limited by computational resources and time, but more extensive tuning could yield better results.
- **Feature Engineering**: Explore additional features such as sentiment scores, word embeddings, or contextual information to enhance model performance
- **Model Deployment**: Consider deploying the best-performing model as a web service or API to allow real-time sentiment analysis of new tweets.