**Data Report**

# **Apple and Google Twitter Sentiment Analysis**

**TABLE OF CONTENT**

**Business Understanding……………………………………………….1**

**Data Understanding……………………………………………………..3**

**Data Preparation.………………………………………………………...5**

**Data Preprocessing..……………………………………………..……..6**

**Modelling………………………………………………………………….8**

**Evaluation, Recommendations, and Conclusion………………….10**

**1. Business Understanding**

**1.1 Business Overview**

Companies such as Apple and Google continuously produce and release new products ranging from smartphones and software updates to smart home devices. With social media platforms like Twitter (currently known as X) becoming key spaces for public expression, users frequently share their opinions, complaints, and praise about these products online.

Understanding these public sentiments provides valuable insight into customer satisfaction, brand perception, and potential areas for improvement. Sentiment analysis using Natural Language Processing offers an efficient, data-driven way to interpret large volumes of social media data and extract actionable insights. This can support marketing, customer experience, and product development strategies

## **1.2 Stakeholder**

* Apple’s and Google’s Product Development and Marketing Teams
* Business Analysts and Data Scientists within both companies
* Decision-makers responsible for customer engagement and brand reputation

## **1.3 Problem Statement**

Given the large volume of audience based data on Twitter, manually analyzing sentiments toward Apple and Google products is time-consuming and inefficient. There is a need for an automated Natural Language Processing model that can accurately classify these tweets into categories such as positive, negative, or neutral, enabling teams to quickly view, analyse, and respond to the publics opinion appropriately.

## **1.4 Objectives**

### **1.4.1 Main Objective**

To develop an effective sentiment analysis model that is able to automatically classify tweets discussing Apple and Google products as positive, negative or neutral

### **1.4.2 Specific Objectives**

* Idenitfy patterns in user sentiments.
* Compare the performances of both binary and multiclass algortihms.
* Provide sufficient recommendations to the companies on how to go about the sentiments.

## **1.5 Research Questions**

1. What is the overall sentiment of Twitter users toward Apple and Google products?
2. Which machine learning approach performs best for classifying tweet sentiment?
3. Can sentiment analysis help identify product or brand perception trends over time?

## **1.6 Success Criteria**

Actionable insights that could inform product or marketing strategies for Apple and Google.

1. **Data Understanding**

**2.1 Importing Relevant Libraries.**

We are going start with getting the relevant libraries that are going to help us understand and inspect our data. We are also importing libraries that will help us with manipulating, visualizing and modelling our data.

This libraries include pandas, numpy, matplotlib, seaborn, sklearn, collections and others.

**2.2 Loading the data.**

We used pandas to download and read our data and covert it into a data frame.

The dataset is a csv file (<https://data.world/crowdflower/brands-and-product-emotions/file/judge-1377884607_tweet_product_company.csv>) comes from data.world where by we are to judge sentiments of people based on there comments. It contains 9092 rows with 3 columns.

**2.3 Initial Exploration and EDA**.

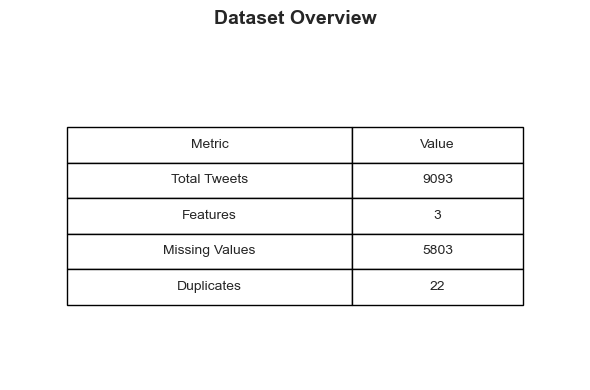
We looked into our data and did exploratory data analysis to see what attributes it has for better understanding.

The dataset includes the following columns:

|  |  |
| --- | --- |
| **Variable/Field Name** | **Description of the variable** |
| tweet\_text | Contains the tweets from people about multiple brands/products. |
| emotion\_in\_which\_tweet\_is\_directed | The product/brand in which the tweet is aimed at |
| is\_there\_an\_emotion\_directed\_at\_a\_brand\_or\_product | The type of sentimet the tweet potrays |

The data is textual in nature

The dataset overview looks like this.



We dived into visualizations to see how various features performed. We noticed that

1. People who had no emtion towards a brand or product accounted for over half the tweets. Positive emotion followed with almost 300 tweets followed by negative emotion tweets and lastly is 'I can't tell' tweets'.

2. ipad had the most tweets which could mean there were a lot of sentiments towards the product. Apple, ipad/iphone app and google followed with other Apple product or service and google where the least tweeted products/brand.

3. The mean length for the tweets is 105 characters long. We have also added a column for ‘tweet\_length’ in our data frame which made our dataframe have 4 columns.

1. **Data Preparation**

**3.1. Data Cleaning**

**3.1.1** We will begin by removing rows with missing values from 'is\_there\_an\_emotion\_directed\_at\_a\_brand\_or\_product' column.

**3.1.2** We removed rows with missing tweet texts

**3.1.3** We droped all duplicated rows in the 'tweet\_text' column but keep the first duplicate.

**3.1.4** We manipulated our ‘tweet\_text’ column so that we have data with significant value to our model and added a new ‘clean\_tweet’ column increasing our data frame to 5 columns

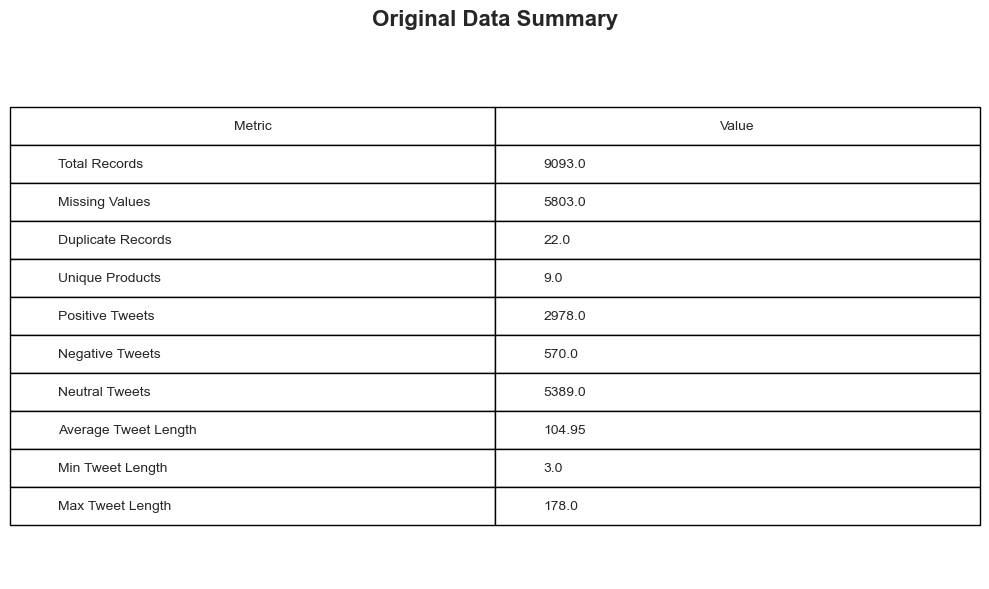
**3.1.5** The columns were renamed for easier reading i.e

* + 'is\_there\_an\_emotion\_directed\_at\_a\_brand\_or\_product'became 'sentiment',
  + 'emotion\_in\_tweet\_is\_directed\_at' became 'product'

**3.2 Advanced Insights**

The most common words is sxsw, link and ipad while the least common words will, popup and this.

The data frame below shows a more in depth look at how original dataset looks like.



1. **Data Preprocessing.**

This section continues from the data cleaning stage and includes tokenization, stop word removal, lemmatization/stemming, rejoining, and vectorization.

**4.1. Cleaning before preprocessing**.

**4.1.1** We imported the cleaned dataset and viewed it on a dataframe.

**4.1.2** Columns that were not required for analysis were dropped from the data frame that is ‘tweet\_text’ and ‘tweet\_length’.

**4.1.3** Checked for missing values and found ‘product’ column to have 5802 missing values, ‘cleaned\_tweet’ column had 1 and ‘sentiment’ had none.

**4.1.4** Dropped the ‘product’ column as it contained many missing values.

**4.1.5** Dropped the row from ‘cleaned\_tweet’ column with a missing value and checked for any missing values.

**4.1.6** Did a value\_count for the ‘sentiment’ column which is

|  |  |
| --- | --- |
| Sentiment | Count |
| No emotion toward brand or product | 5388 |
| Positive emotion | 2978 |
| Negative emotion | 570 |
| I can’t really tell | 156 |

It was evident that the sentiment classes are imbalanced, with a higher number of “No emotion toward brand or product” and “Positive emotion” tweets compared to “Negative emotion” and “I can’t tell.” This imbalance may affect model performance, as the classifier could become biased toward the majority classes. Therefore, balancing techniques such as stratified sampling or SMOTE were considered during the modeling stage.

4.2 **Encoding the Sentiment Column.**

**4.2.1** We merged 'No emotion toward brand’ class and 'I can't tell' class into one 'Neutral' class as they more or less explained the same thing thus bringing the class numbers to 3. The classes were then assigned numerical values for easier analysis. The classes became target variables with assigned values as follows:

|  |  |
| --- | --- |
| Negative emotion | 0 |
| Positive emotion | 1 |
| Neutral | 2 |

**4.2.2** Dropped the ‘sentiment’ column as it was converted to the ‘target’ column for analysis

**4.3 Preprocessing tweets.**

**4.3.1** Imported the nltk library which is used for preprocessing tweets for analysis

**4.3.2** Defined a function that we ued to preprocess the tweets by using nltk process like

* Tokenize – to break down the tweets to smaller units (words).
* Stop words – to get rid of words like the, and etc.
* Lemmatize – to reduce words to there basic forms

The function was then applied and column called ‘tweets’ was returned.

**4.3.3** The ‘cleaned\_tweet’ was dropped ahead of vectorization as we now have tokenized ‘tweet’ column.

1. **Modelling**

The modelling will be conducted in two main parts:

1. Binary classification:

    - Positive emotion

    - Negative emotion

3. Multiclass classification, with 3 categories:

    - Positive emotions

    - Negative emotions

    - Neutral emotion

**5.1 Binary modelling**

- In this section, the data was filtered to include only the positive and negative emotion classes.

- We then performed vectorization, handle any class imbalance, and train multiple models, including Logistic Regression, Multinomial Naive Bayes, and a Support Vector Machine (LinearSVC) model, which is particularly well-suited for text and high-dimensional sparse data.

- We looked at the distribution of our targets with 1 having 2978 inputs and 0 having 570 inputs

- We instantiated a vectorizor and proceeded to define our X and y where X became our vectorized tweets and y our targets.

- We split our data into a 20% test size with a random state=42 and stratified y.

- Due to the class imbalance, we used SMOTE to balance out the training data for unbiased outcome.

**5.1.1 Training a Logistic Regression model**

- After Instantiating our model, training and predicting our test size, the model gave an accuracy of 84.7%

**5.1.2 Training a Multinomial Naïve Bayes**

This model gave us an accuracy of 85.3%

**5.1.3 Training a Support Vector Machine model**

This model gave us a 89.1% accuracy

Confusion matrix plots were made to better visualize our outcome.

**5.1.4 Binary Model performance Comparison.**

From the above matrix confusion matrix, we can conclude that:

1. The linear SVM produced the best model out of the three as it was able to predict accurately 89% of the time.

2. The SVM model had a higher false positive (Type 1 error) reading compared to Multinomial Naive Bayes, but a lower false negative(Type 2 error)

3. Logistic regression perfomed the poorest out of the 3 with an accuracy score 84.7 %. Multinomial had an accuracy score of 85.3%.

**5.2 Multiclass Modelling**

In this section, we’ll build models to classify tweets into three sentiment categories:

- Positive emotion (1)

- Negative emotion (0)

- Neutral (2) → merged from “No emotion toward brand or product” and “I can’t tell”.

Using the same process as binary modelling, we instantiated a model, defined X and y, split our data into a 20 % test size with random\_state = 42 and a stratified y and finally dealt with the class imbalance using SMOTE

**5.2.1 Training a Logistic Regression Model**

- After fitting and predicting the model, The accuracy score for this model was 67.5%

**5.2.2 Training a Multinomial Naïve Bayes.**

- The accuracy score was 62% for this model

**5.2.3 training a Support Vector Machine Model**

- Accuracy score was 69.1%

Confusion matrix plots were made for the 3 multiclass models

**5.2.4 Multiclass Model Perfomance Evaluation.**

According to the above, we can see that:

1. The SVM model still did well with an acccurcy score of 69.1 % able to predict the targets more accurately.

2. Here, with the inclusion of a third target, the logistic regression model did better with an accuracy score of 67.5 % compare to Multinomial Naive Bayes which had 62% accuracy.

3. SVM model and logistics had high false positives with Multinomial Naive Bayes has high false negatives.

**5.2.5 Multiclass Hyperparameter Tuning On SVM Model**

The SVM model proved to be the better perfoming model so we used hyperparameter methods like GridSearchCV for tuning to see if it perfoms better.

A confusion matrix was plotted to se visualize our outcome.

**5.2.6 Hyperparameter Tuning Performance Review**

After tuning our Support Vector Machine model, the training accuracy is 85% with testing accuracy still 69%. Most notable is Class 1 target errors are heavily biased towards being misclassified as Class 2 213 times and Class 2 target errors are heavily biased towards being misclassified as Class 1 240 times.

## **6.0 Evaluation, Recommendations, and Conclusion**

## **6.1 Overview:**

This section evaluated the performance of the sentiment classification models and provides recommendations for improvement and future work.

The project involved

* Binary sentiment classification (positive vs. negative)
* Multi-class sentiment classification (negative, neutral, positive)
* Feature engineering: TF-IDF vectorization for text representation
* Balancing techniques: SMOTE to handle class imbalance
* Models tested: Logistic Regression, Naive Bayes, and LinearSVC
* Hyperparameter tuning: Performed using GridSearchCV to optimize model performancentment

To support accurate model learning, TF-IDF vectorization was used for feature extraction, while SMOTE addressed class imbalance by synthetically generating samples for underrepresented classes. The models tested included Logistic Regression, Naive Bayes, and LinearSVC, with hyperparameter tuning performed via GridSearchCV to optimize model performance.

### **6.2 Evaluation**

Both binary and multi-class classification models were implemented to analyze sentiment from Apple and Google tweets. TF-IDF ensured strong feature representation, and SMOTE improved recall for underrepresented sentiments.

#### **Model Performance Summary**

Model Accuracy Precision Recall F1-score

Logistic Regression 0.847 0.84 0.84 0.84

Baseline Naive Bayes 0.853 0.85 0.85 0.85

Baseline LinearSVC (Tuned) 0.690 0.70 0.69 0.69 Tuned

#### **Key Findings**

Best Performing Model: Tuned LinearSVC achieved the most balanced performance across all metrics.

Impact of SMOTE: Improved recall for minority classes, especially negative sentiments.

Common Misclassifications: Neutral tweets were often confused with positive ones due to subtle tone or sarcasm.

#### **Confusion Matrix Insights**

False Positives: Neutral tweets predicted as positive.

False Negatives: Positive tweets predicted as neutral.

Neutral sentiments proved the hardest to classify, largely because of linguistic ambiguity and mixed emotions typical in social media discourse.

### **6.3 Recommendations**

Data Enrichment: Expand the dataset with recent and diverse tweets to improve generalization across slang, emojis, and evolving language patterns.

Advanced Text Representations: Implement contextual word embeddings (e.g., Word2Vec, GloVe) or transformer-based models (e.g., BERT, RoBERTa) for deeper semantic understanding.

Contextual Awareness: Use models capable of detecting sarcasm and nuanced emotion through contextual learning.

Real-Time Dashboard: Integrate the tuned LinearSVC into a live monitoring system for marketing and customer engagement insights.

Aspect-Based Sentiment Analysis: Extend the analysis to evaluate sentiment toward specific product features such as camera quality, battery life, or performance.

### **6.4 Summary of Modelling Results**

Binary Classification: Naive Bayes achieved a higher accuracy (85.3%), outperforming Logistic Regression (84.7%) with strong generalization between positive and negative sentiments. SVM had highest accuracy at 89%.

Multi-Class Classification: The tuned LinearSVC achieved 85% training and 69% testing accuracy, showing balanced precision and recall. Logistic Regression and Naive Bayes followed at 67.5% and 62%, respectively.

Trends: Neutral tweets caused the most misclassifications due to ambiguous tone and informal language such as slang and emojis.

### **6.5 Limitations**

Minority Class Prediction: Despite SMOTE, predicting neutral sentiment remained challenging due to overlapping linguistic patterns.

Model Limitations: TF-IDF combined with classical ML models (Logistic Regression, Naive Bayes, LinearSVC) lacks semantic depth, limiting contextual understanding.

Linguistic Complexity: Elements like sarcasm, abbreviations, and emojis reduce interpretability for traditional ML approaches.

### **6.6 Conclusion**

Binary Classification: Naive Bayes delivered the highest accuracy (85.3%), slightly outperforming Logistic Regression (84.7%).

Multi-Class Classification: Tuned LinearSVC achieved 85% training and 69% testing accuracy, balancing precision and recall effectively.

#### **Key Takeaways**

Model Strengths: Naive Bayes excelled in binary classification; LinearSVC performed best in multi-class tasks.

Accuracy Trends: Positive and negative sentiments were well captured, while neutral sentiments remained ambiguous.

Objective Fulfillment: Models successfully automated sentiment classification, generating data-driven insights into brand perception.

#### **Future Work:**

Integrate contextual embeddings or transformer architectures for deeper semantic learning.

Deploy models in real-time systems for continuous sentiment tracking.

Expand to aspect-based sentiment analysis for targeted business insights