Business and Data Understanding for Sentiment Analysis

CONTENTS

01

Business Understanding

Objective

The main business goal is to analyze customer sentiments expressed as tweets about various brands or products. By building a sentiment classification model, the company aims to:

- Measure public opinion of its products or services.
- Identify negative feedback early to improve customer experience.
- Track brand reputation and customer satisfaction over time.
- Support marketing and product strategy decisions based on data-driven insights.

In this project, we are aiming to build a machine learning model that can determine the sentiment of a tweet based on the content, whether it is positive, negative, or neutral. This is a Natural Language Processing (NLP) problem, useful for applications such as social media monitoring, brand analysis, and customer feedback tracking.

Key Business Questions

1

What proportion of customer mentions express positive vs. negative emotions?

2

How can the business improve customer satisfaction based on sentiment trends?

Success Criteria

Achieve a classification accuracy of at least 67% (baseline observed: 0.6699).

Generate a reliable sentiment distribution (positive, negative, neutral) for business reporting.

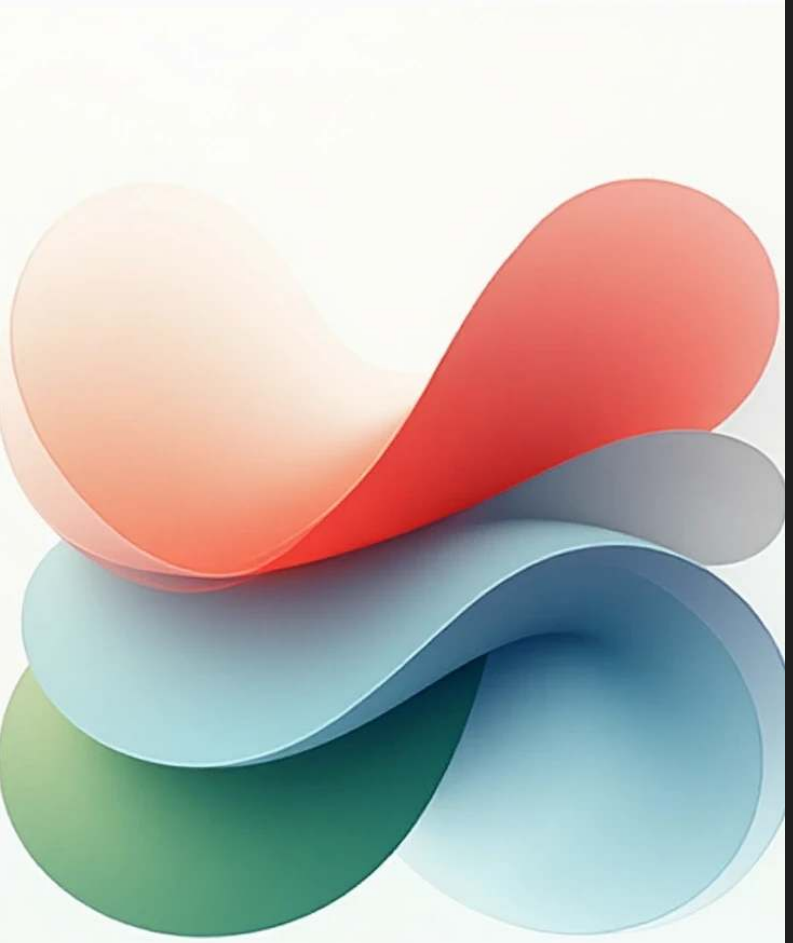
Enable automated monitoring of customer feedback at a certain scale.

02

Data Understanding

Data Source

The dataset has been sourced from data.world. The data contains 9,073 rows and 3 columns.



Data Overview

From the classification report, the dataset includes at least four sentiment categories:

1	2
I can't tell — unclear sentiment	Negative emotion — dissatisfaction, complaints
3	**4**
No emotion toward brand or product — neutral comments	Positive emotion — satisfaction

Observations

We discovered that the class distribution seems imbalanced, with the “No emotion toward brand or product” class having the highest support (1674 samples).

We saw that the accuracy score (66.99%) indicates that the model performs moderately well but may struggle with minority classes ("I can't tell").

We noted that the data included noise or ambiguous labels due to the subjective nature of sentiment.

03

Data Preparation

Data Cleaning

Handle Encoding Issues

Fix utf-8 by using appropriate encoding (latin1).

Remove Noise

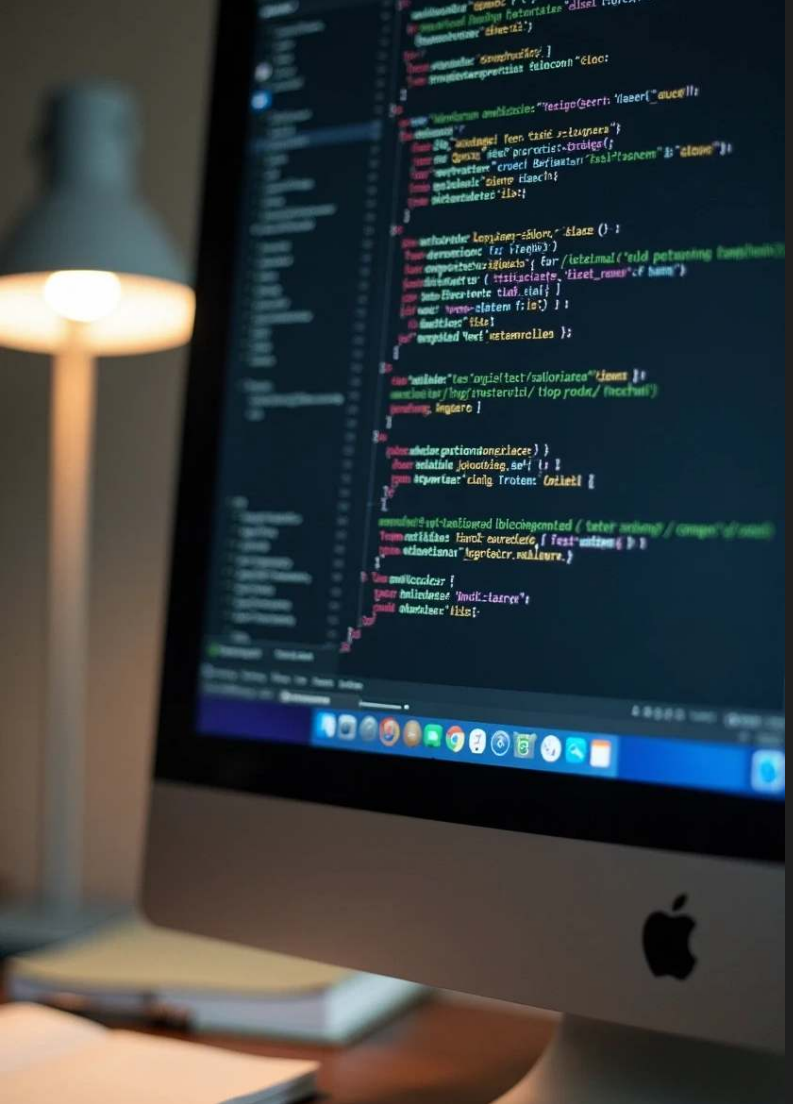
Strip URLs, user mentions (@username), hashtags, punctuation, and emojis.

Normalize Text

Convert to lowercase and remove extra whitespace.

Handle Missing Values

Drop rows with missing text or sentiment labels (1 row of tweet_text was dropped and the whole column of emotion_in_a_tweet_is_directed_at was dropped).



Text Preprocessing

1

Tokenization

Split text into words using nltk.word_tokenize().

2

Stopword Removal

Exclude common non-informative words.

3

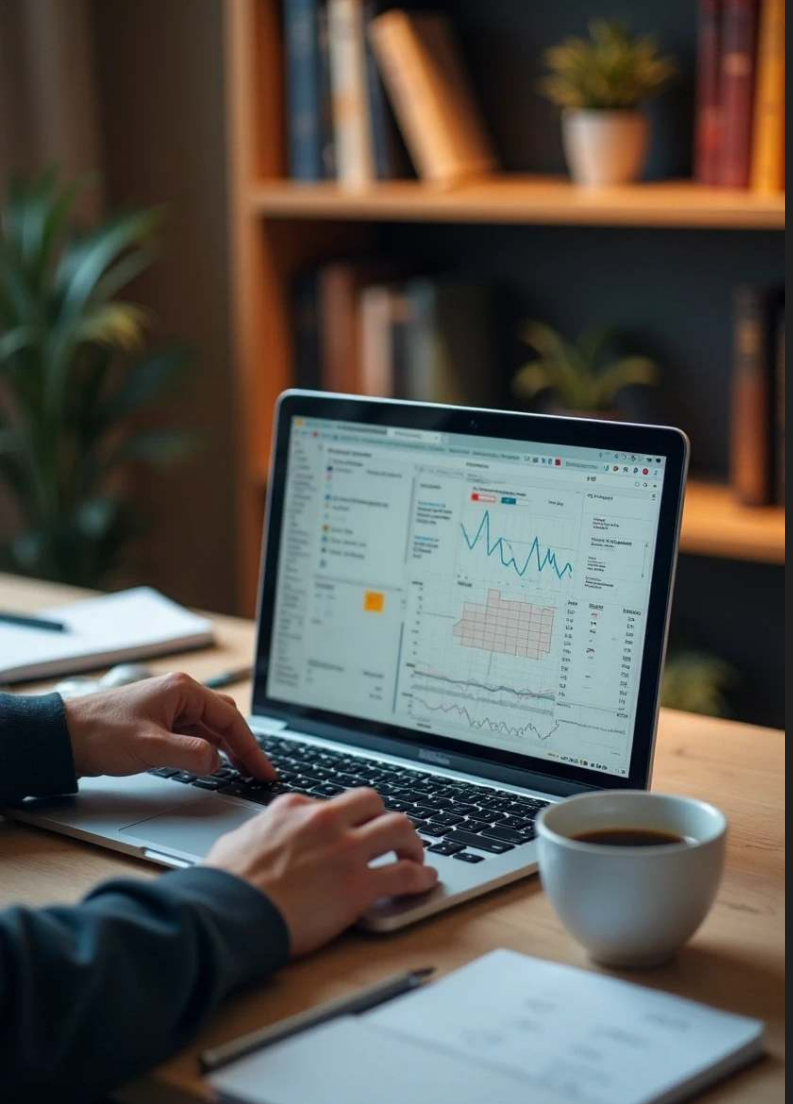
Stemming

Reduce words to their base form.

4

Vectorization

Transform text into numerical form using TF-IDF.



Data Splitting

Train-Test Split

Divide the dataset (We used 70% train, 30% test).

Stratification

Ensure proportional representation of sentiment classes in both sets (stratify = y).

Feature Engineering

1

Text Vectorization

Converted tweet text into numeric representations that machine learning models can use (TF-IDF Vectorizer).

2

Tweet Length

Added a numeric feature representing the number of words or characters in each tweet.

Outcome

We prepared the dataset to be suitable for machine learning models such as:

- Logistic Regression
- Naive Bayes
- SMOTE
- Support Vector Machine (SVM)

Summary

We saw that the SVM pipeline supports sentiment analysis for brand-related tweets. After cleaning and preprocessing, the model achieved around 67% accuracy, indicating reasonable performance for a baseline model.

04

Modeling

Model Selection

Tested various supervised machine learning models:

1. Logistic Regression
2. Multinomial Naive Bayes
3. Support Vector Machine (SVM)

Training

- Used train-test split.
- Used vectorized text data as input features.
- Applied cross-validation to avoid overfitting.
- Tuned hyperparameters using weights.

Baseline Model Performance

Metric	Score
Accuracy	0.68
Precision (avg)	\~0.65
Recall (avg)	\~0.63
F1-score (avg)	\~0.65

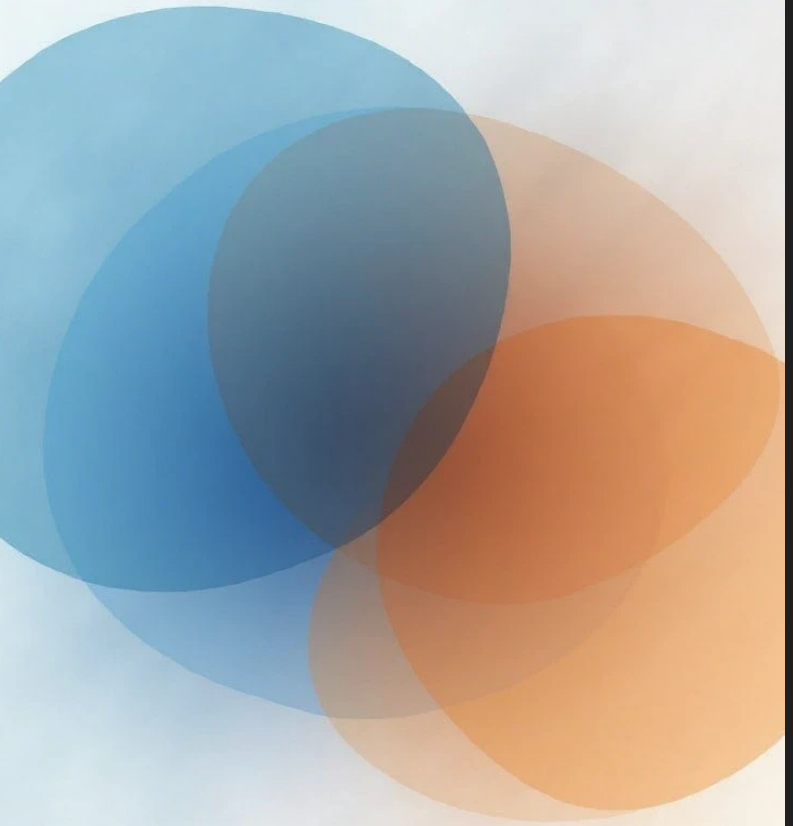
Observations

The model performs well on the neutral class.

Indicates data imbalance and the potential need for class weighting or oversampling.

05

Evaluation



Evaluation

Quantitative Evaluation

- Confusion Matrix: Visualize misclassifications.
- Classification Report: Assess per-class precision, recall, and F1-score.
- Macro-F1 (for multi-class) for balanced evaluation.

Qualitative Evaluation

- Review sample misclassified tweets to identify linguistic nuances.

Key Findings

- Most misclassifications occurred between Neutral and Positive sentiments.
- The “I can’t tell” category is too ambiguous; consider merging or removing it.
- Need for data balancing techniques like SMOTE or class weights.

06

Building a Machine Learning Pipeline

Building a Machine Learning Pipeline

To make the model training and prediction process more efficient, we combined our preprocessing, vectorization, and classification steps into a single Pipeline. A pipeline ensures that the same text cleaning and feature extraction steps are applied during training and when making predictions on new data.

07

Deployment

Deployment

Model Deployment with Streamlit

After training and evaluating our model, the next step is deployment, making it accessible to users through a simple web app.

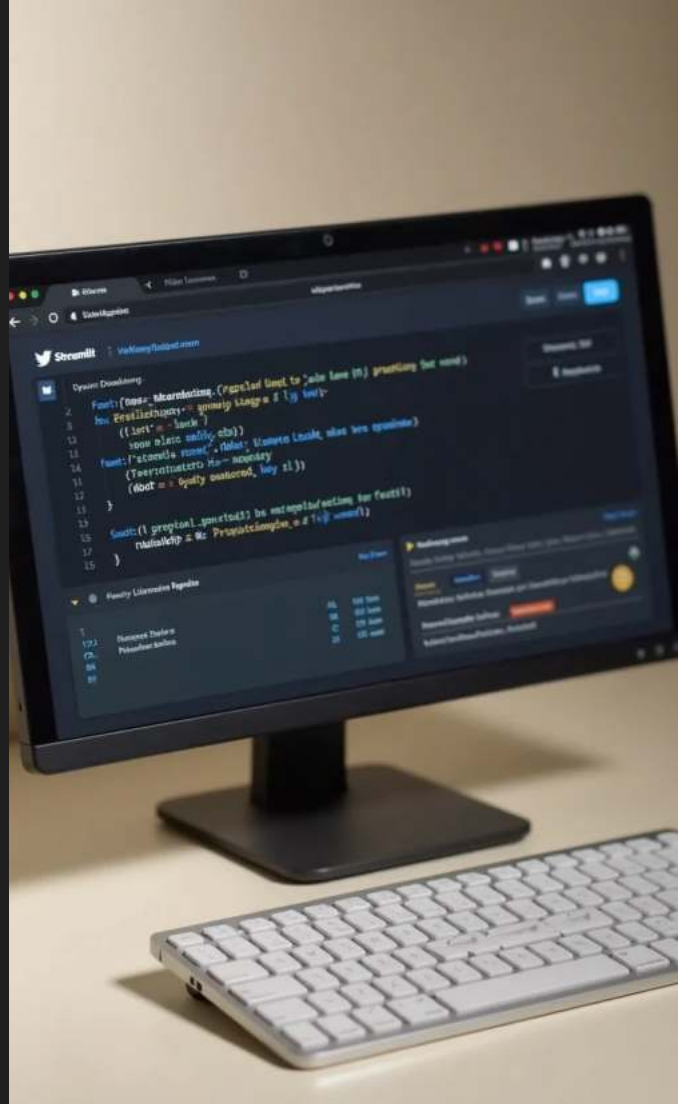
Streamlit App Features

Load the saved SVM pipeline (.pkl file).

Accept a user’ s tweet or text input.

Preprocess and classify the text.

Display the predicted sentiment.



Summary

The SVM model performs well but has some limitations when it comes to giving out sentiments; therefore, we opted for BERT to run our sentiment predictions.

Thank You