# Sentiment Analysis for Arabic Company Reviews Using AWS

## Project Overview

This project focuses on real-time Sentiment Analysis for Arabic Company Reviews using AWS.   
The objective is to classify company reviews into Positive, Neutral, or Negative sentiments by leveraging Machine Learning (ML) and Natural Language Processing (NLP) techniques.

## Objectives

✔ Develop an end-to-end sentiment analysis pipeline on AWS.  
✔ Classify Arabic reviews into Positive, Neutral, or Negative categories.  
✔ Automate data ingestion, preprocessing, model training, and deployment.  
✔ Deploy a real-time API for sentiment prediction.

## Value Proposition

This solution benefits multiple stakeholders:  
- Companies & Business Owners (e.g., Talabat, Venus): Track service quality to improve customer satisfaction.

- Investors & Business Analysts: Evaluate customer sentiment trends to make informed investment decisions.  
  
 - Market Researchers: Gain insights into consumer behavior and service trends in the Middle East.

## Expected Outcomes

✔ Enhanced Customer Experience: Identifying and addressing pain points such as slow delivery and app issues.  
✔ Higher Profitability: Businesses can focus on high-value services based on customer feedback.  
✔ Innovation: Detecting gaps in services and opportunities for expansion.

## Step 1: Data Collection & Storage

**Arabic Company Reviews Dataset**

The dataset contains **over 40,000 Arabic company reviews**, structured as follows:

* **Review Text (التقييم النصي)** – The actual review in Arabic.
* **Company Name (اسم الشركة)** – The name of the reviewed company.
* **Rating (التقييم)** – A numerical rating associated with the review.

## **Step 2: Data Exploration**

### Exploratory Data Analysis (EDA) Report for Arabic Company Reviews Dataset

**Dataset Overview:** The dataset contains **40,046** reviews from customers about various companies, written in Arabic. Each review is labeled with a **sentiment score** and is associated with a specific company.

**Columns:**

* review\_description (Text) – The actual customer review.
* rating (Numeric) – Sentiment score:
  + **1** → Positive review
  + **0** → Neutral review
  + **-1** → Negative review
* company (Text) – The company being reviewed.

**Sentiment Distribution**

* The dataset has **a mix of positive, neutral, and negative reviews**.
* A visualization of **rating distribution** would help in identifying class imbalances.

📌 **Potential Issues:** If one sentiment class (e.g., positive reviews) dominates, model performance may be biased.

**Data Problems:**

* **Conflict reviews**: there are 89 reviews belong to two classes -1 and 1
* Reviews with **emojis**
* A pie chart with numbers and a red circle

  AI-generated content may be incorrect.**Empty and Duplicated** reviews after keeping only letters and spaces = 1820 empty and 691 duplicated reviews
* **Rating distribution**: Class imbalances class
* **Imbalances company reviews**

A screenshot of a computer

AI-generated content may be incorrect.

## **Step 3: Data Cleaning**

**Removed Unnecessary Columns:** Dropped Unnamed: 0, which was an index column.

**Checked for Duplicates:** Found **28 duplicate reviews**, which may need to be removed.

**Checked for Missing Values:** Further analysis is required to determine if there are missing reviews.

## **Step 4: Data preprocessing**

**🔹** **Handling Missing Values:**  
Null values were removed from the dataset, as their presence was minimal and did not significantly affect the overall data size.

**🔹 Removing Duplicates:**  
Duplicate records were dropped due to their limited number, ensuring data consistency without impacting the dataset's volume.

**🔹** **Filtering "Natural" Class with Zero Labels:**  
Rows labeled as "natural" with a value of 0 were excluded from the dataset. These instances were fewer than 2,000 and contributed to class imbalance, which could negatively affect the model's performance.

**🔹** **Filtering out non-Arabic words:**

Because this project focuses on Arabic , we kept only the Arabic text using regular expression

**🔹 Stop Words**

Arabic stop words often carry **semantic importance**, and their removal can unintentionally distort meaning. For instance:

* **Original:** *"لا أحب هذا المنتج"* (I do **not** like this product)
* **After Removing All Stop Words:** *"أحب هذا المنتج"* (I **like** this product)
* **Issue:** The sentiment is **completely reversed** after removing *"لا"*.

To prevent such distortions, we retain critical stop words that contribute to **negation, contrast, and logical structure**, including:

* **Negation:** *"لا"*, *"ليس"*
* **Contrast & Exception Handling:** *"لكن"*, *"ولكن"*, *"إلا"*

By preserving these words while filtering out less significant stop words, we ensure that sentiment analysis remains **accurate and contextually meaningful**.

 => 

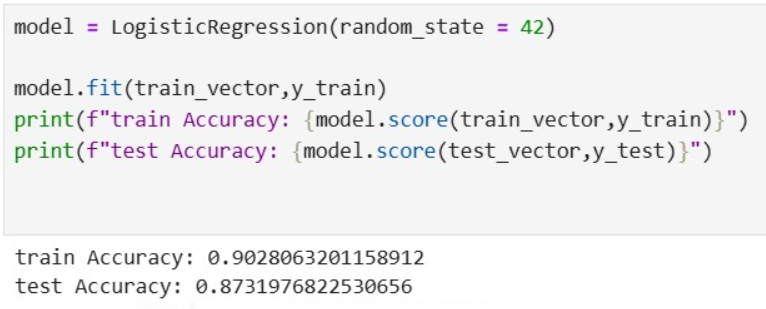
 => 

However, we soon realized that the standard Arabic stop word list did not cover many commonly used words in Arabic dialects, particularly Egyptian dialect, which dominated the dataset. Since the data primarily consists of colloquial (non-standard) Arabic reviews, we encountered many frequent but non-standard words that were not filtered out.

A group of small red and black letters

AI-generated content may be incorrect.To address this issue, we manually compiled a list of the most common dialect-specific stop words that appeared repeatedly in the dataset, based on our linguistic knowledge and practical observations. Additionally, we leveraged ChatGPT to suggest other possible dialectal stop words we might have missed, ensuring a more comprehensive and accurate stop word list tailored to our data.

**After applying the initial basic preprocessing steps, we experimented with several models, including Logistic Regression. The results showed promising accuracy levels. However, in our pursuit of higher precision and improved overall performance, we decided to continue refining the preprocessing pipeline to enhance the model’s effectiveness and reliability.**

****

**So we started to find out more about the problems in the data.**

**🔹 Removing Punctuation**

In sentiment analysis, punctuation typically does not contribute significantly to determining sentiment polarity. Since our goal is to classify reviews as **positive, negative, or neutral** rather than analyzing intensity or emotional range removing punctuation can help simplify text processing without losing valuable sentiment cues. For example, while an exclamation mark (!) may indicate emphasis, it does not alter the fundamental sentiment classification.

**🔹 Diacritics (Harakat)**

In Arabic, diacritics (known as *Harakat*) such as َ (Fatha), ُ (Damma), ِ (Kasra), ّ (Shadda), and ْ (Sukun) are optional pronunciation guides added to letters. They do not typically affect the *semantic meaning* of words in modern text usage, especially in informal contexts such as reviews, comments, or social media posts.

Why we did it:

* Most Arabic text data found online is unvoweled (i.e., without diacritics).
* Including diacritics can lead to artificial variations of the same word. For example:
  + "كِتاب" (with Kasra) and "كتاب" (without diacritics) would be treated as two different tokens by the model.
* Removing diacritics helps reduce vocabulary size and improve consistency, which is essential for downstream NLP tasks like lemmatization and tokenization.

**A close-up of a computer code

AI-generated content may be incorrect.**

** => **

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**🔹 Emojis**

**A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer code

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.**

Since most emojis can be misleading, we have decided to remove them to prevent any potential distraction for the model. Additionally, emojis have minimal impact on our task. Furthermore, the number of rows containing only emojis is relatively small compared to the overall dataset.

****

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**🔹 Num2Word**

We convert all numerical values, whether in Arabic or English, into their corresponding words.

****



**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**🔹 Normalizing Text**

Arabic is a rich language with many variations of characters that can represent the same phonetic sound. For example:

* The letter "أ" (Alef with Hamza) can be written as "ا" (Alef without Hamza).
* The letter "ى" (final form of "Yaa") is often used interchangeably with "ي" (Yaa).
* The letter "ة" (Taa Marbouta) can be replaced by "ه" (Haa) in some contexts.

**A group of black text

AI-generated content may be incorrect.Why we did it:**  
Normalization is crucial because these variations do not affect the meaning of words but can lead to discrepancies when performing text analysis. By normalizing the text, we:

* **Reduce redundancy** in the data, ensuring that different forms of the same word are treated as identical.
* A computer screen shot of text

  AI-generated content may be incorrect.**Improve consistency** for tasks like tokenization, lemmatization, and model training, thus helping the model to generalize better.

**🔹 Removing Word Elongation**

In Arabic, especially in informal contexts or dialects, words often have repeated characters, a phenomenon known as "elongation." For example, the word "جداااااا" (meaning "very") may appear with repeated characters, which is common in online communication and text messaging. This elongation does not carry any additional meaning and can lead to inconsistencies in text analysis or machine learning tasks. Therefore, removing word elongation helps standardize the text.

**Why we did it:**  
Elongation can lead to noise in natural language processing (NLP) tasks, including:

* **Inconsistent tokenization**: Words like "جداااااا" should be treated as "جدا".
* **Model inefficiency**: Models may misinterpret the meaning of words due to unnecessary character repetition.

By removing elongation, we make the text more consistent and easier to process.

A screen shot of a computer code

AI-generated content may be incorrect.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**🔹 Lemmatization**

In our Arabic Sentiment Analysis project, lemmatization was a critical step in the text preprocessing pipeline. Lemmatization involves reducing words to their base or dictionary form (lemma), which improves model generalization and reduces vocabulary size, an essential step for building accurate and reliable machine learning models.

* **Why We Chose Qalsadi?**

**1. Native Support for Arabic Language:**

* **Qalsadi** is specifically designed for Arabic morphological analysis. It leverages the **root-based structure of Arabic**, handling inflections, prefixes, suffixes, and grammatical variations more accurately than generic libraries.
* It can, for example, convert words like “يكتبون” (they write) to the root “كتب” (write), preserving the semantic meaning.

**2. High Linguistic Accuracy:**

* Qalsadi effectively handles **complex Arabic grammar**, including verb conjugations, noun pluralization, feminine/masculine forms, and pronoun attachments.
* It accounts for **diacritics, root derivations**, and more—crucial aspects that generic NLP tools usually miss.

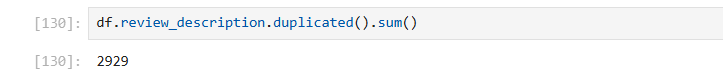
**3. Seamless Integration with Arabic NLP Pipeline:**

* Qalsadi works well with other preprocessing steps such as **diacritic removal**, **text normalization**, and **stopword filtering** all tailored for Arabic language needs.
* This synergy makes it a better fit for end-to-end Arabic text processing.

**Why We Didn’t Use spaCy Despite Its Speed?**

|  |  |  |
| --- | --- | --- |
| **Factor** | **Qalsadi** | **spaCy** |
| Native Arabic Support | Excellent | Very limited |
| Verb & Morphological Accuracy | High | Poor for Arabic |
| Morphological Rules | Rule-based | Not Arabic-aware |
| Speed | Slower | Faster |
| Suitability for Dialect/Informal Text | Customizable | Not practical |

After applying lemmatization, it results many duplicated reviews so, we removed them.



**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**🔹 Subjectivity Prediction with Pre-trained Model**

At first we found some reefs that do not carry an opinion but information so we decided to do the Subjectivityscore on the data.

The purpose of this section of the code is to predict whether a given text is "subjective" or "objective". This is done using a pre-trained model designed for sequence classification in the Arabic language. The model's task is to classify whether the content of a text represents a personal opinion or subjective viewpoint (Subjective) or factual information (Objective).

Subjectivity detection is essential in many natural language processing (NLP) applications, such as sentiment analysis, opinion mining, and content filtering. It allows us to classify whether the text expresses personal opinions (like in reviews, social media posts) or objective statements (like news articles, encyclopaedic content). In this case, the model used is specifically trained for the Arabic language, making it highly suitable for Arabic text analysis.

A close up of a text

AI-generated content may be incorrect.A computer screen shot of a program

AI-generated content may be incorrect.The code loads a pre-trained transformer model for sequence classification using the transformers library by Hugging Face.



**🔹 Embedding:**

During the embedding phase, the team was divided into three groups to explore different approaches.

* The first group focused on traditional embedding techniques, specifically experimenting with **TF-IDF**.
* The second and third groups explored deep learning-based embeddings, utilizing representations generated by large language models (LLMs). They experimented with **AraBERT** and **UBC-NLP/MARBERT** embeddings.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model Name** | **Language Support** | **Embedding Quality** | **Best Use Case** | **Dialect Support** | **Ease of Use** |
| asafaya/bert-base-arabic | Arabic (Fusha) | Very good | Classification, sentiment, clustering | None | Easy |
| aubmindlab/bert-base-arabertv2 | Arabic (Fusha + light dialects) | Excellent | Sentiment, classification, embeddings | Limited | Easy |
| UBC-NLP/MARBERT | Arabic Dialects | Excellent | Social media, dialect-rich texts | Strong | Easy |
| CAMeL-Lab/camelbert-ca | Arabic (Fusha) | Excellent | NLP tasks like POS, NER, classification | None | Easy |
| xlm-roberta-base | 100+ languages | Good in Arabic | Cross-lingual tasks, semantic search | Okay | Easy |
| sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2 | 50+ languages | Very high | Semantic search, similarity, retrieval | Decent | Very easy |
| tiiuae/falcon-rw-1b | Multilingual (some Arabic) | Under testing | General LLM tasks, not embedding-focused | Basic | Needs setup |

Following the embedding stage, the teams agreed to evaluate the performance of four different models on all three embedding types. The goal was to compare the results and select the configuration that achieved the highest accuracyfor final implementation.

**5-Text Classification**

We conducted a comparative analysis of feature extraction methods and classification algorithms to determine the optimal approach for our dataset.

**Classification Algorithms**

* **Logistic Regression**: A linear model suitable for binary and multiclass classification, known for its simplicity and interpretability.
* **SVM (Support Vector Machine)**: A robust algorithm that maximizes the margin between classes, effective for high-dimensional data.
* **Random Forest**: An ensemble method that leverages multiple decision trees to improve robustness and reduce overfitting.
* **XGBoost**: A gradient boosting algorithm optimized for performance, capable of handling complex patterns in data.
* **LSTM (Long Short-Term Memory)**: A recurrent neural network designed for sequential data, excelling in capturing long-term dependencies for time-series or text classification.

•  **GRU (Gated Recurrent Unit):** A simplified variant of LSTM, offering comparable performance with faster training for sequential data classification.

**Findings:**

Initially, we adopted the TF-IDF embedding technique because it yielded nearly the same accuracy (87%) as advanced deep learning methods, such as using AraBERT & MABERT embeddings with models like SVM, Random Forest, LSTM, GRU and XGBoost. This made it a strong candidate due to its simplicity and low computational cost. In both approaches, we consistently used **Logistic Regression** as our classification model, as it provided stable and interpretable results.

However, through further qualitative analysis, we discovered a key limitation of the TF-IDF approach: its inability to effectively capture semantic nuances in certain sentence structures particularly those involving **negation**. For example, a sentence like *"الأكل مش حلو"* (which expresses a negative sentiment) was misclassified by the TF-IDF model because it treats words independently and lacks contextual understanding.

When we tested sentence embeddings generated using language models (LLMs), we found that these models were better at capturing the overall meaning of such sentences, correctly identifying the sentiment in cases involving negation or subtle linguistic cues even when using the same **Logistic Regression** classifier.

Based on this insight, we concluded that although the accuracy improvement with embeddings was not drastic, the **quality and reliability** of predictions—especially for linguistically complex cases—were better with contextual embeddings. As a result, we favored embeddings from LLMs for improved generalization and sentiment comprehension, while keeping Logistic Regression as our preferred classification method.

## Resources

GitHub link about preprocessing for the data: <https://github.com/saobou/arabic-text-preprocessing/blob/master/Preprocess.ipynb>

How To Solve Alef With and Without Hamza Problem: <https://ahmadessamdev.medium.com/arabic-case-insensitive-in-database-systems-how-to-solve-alef-with-and-without-hamza-problem-c54ee6d40bed>

Arabic NLP — How To Overcome Challenges, Tutorials In Python & 9 Tools/Resources Including Large Language Models (LLMs): <https://spotintelligence.com/2023/10/29/arabic-nlp/?utm_source=chatgpt.com>

Paper about Arabic Natural Language Processing: Challenges and Solutions: <https://www.researchgate.net/publication/206006010_Arabic_Natural_Language_Processing_Challenges_and_Solutions>

[Arabic Survey](https://github.com/iwan-rg/ArabicSurvey): <https://github.com/iwan-rg/ArabicSurvey>