FaithWambugu / **NLP-Project**

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

NLP-Project Public

12 Branches 0 Tags Go to file t Go to file Add file Code

FaithWambugu Delete sentiment_analysis2.ipynb 89d884f · now

Data	Added folders	12 hours ago
Images	Added Presentation and i...	7 hours ago
Notebook PDF	Final Touches	9 minutes ago
Presentation	Added folders	12 hours ago
.gitignore	Configured gitignore to e...	last week
README.md	Revise README for clarit...	11 hours ago
environment.yml	Update environment.yml ...	2 days ago
requirements.txt	Merge pull request #14 fr...	10 hours ago
requirements_upd...	Add updated requiremen...	11 hours ago
sentiment_analysis...	Add files via upload	3 minutes ago
sentiment_app.py	Fix: update Streamlit app ...	14 hours ago

About

No description, website, or topics provided.

Readme
Activity
0 stars
0 watching
1 fork

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Contributors 6

README

🍏 Apple & Google Tweet Sentiment Analyzer 🍎

Real-Time Sentiment Monitoring for Tech Distributors

This project leverages **Transfer Learning** with the **DistilBERT** model to provide third-party technology distributors with critical, real-time insights into public sentiment towards Apple and Google products. The primary goal is to establish a robust alerting system by maximizing the detection of negative customer feedback.

Our final model achieved an exceptional 90 **Recall** on the Negative sentiment class, ensuring that the highest-risk tweets are flagged immediately for business action.

💡 Business Context & Motivation

Third-party distributors operate in a high-stakes, competitive environment where product perception dictates sales and inventory strategy. They currently lack accessible tools for automated, large-scale sentiment analysis, leaving them vulnerable to market shifts and slow to react to customer dissatisfaction.

This solution bridges that gap by transforming unstructured social media data into **actionable intelligence**, enabling distributors to:

Languages

- **Jupyter Notebook** 99.8%
- **Python** 0.2%

Suggested workflows

Based on your tech stack

Python Package using Anaconda Configure

Create and test a Python package on multiple Python versions using Anaconda for package management.

Publish Python Package Configure

Publish a Python Package to PyPI on release.

SLSA Generic generator Configure

Generate SLSA3 provenance for your existing release workflows

More workflows Dismiss suggestions

- **Prevent Missed Warnings:** Proactively address potential product issues or widespread customer frustration.
- **Optimize Inventory:** Align stocking decisions with real-time brand perception.
- **Inform Marketing:** Adjust campaigns and messaging based on evolving public opinion.

Key Technical Features

The solution employs a hybrid modeling approach to serve both alerting and strategic analysis needs:

1. **Binary Classification (Alerting Core):**
    - **Goal:** High Recall for the Negative class (0.90 achieved).
    - **Model: DistilBERT** fine-tuned on the Negative vs. Positive classes.
2. **Multiclass Classification (Strategic Trends):**
    - **Goal:** Contextual insight into all sentiment types (Negative, Neutral, Positive).
    - **Model: DistilBERT** fine-tuned on the three sentiment classes.
3. **Advanced Handling of Imbalance:**
    - **Technique:** Combination of **Data Augmentation** (Synonym Replacement) and a custom **Focal Loss** function were implemented to effectively address the severe class imbalance, surpassing the performance of traditional SMOTE/Class Weights.

📊 Performance Summary

The Deep Learning approach decisively outperformed the classical Machine Learning models (Logistic Regression, Random Forest, XGBoost) across all key metrics.

Model	Classification Task	Negative Recall	Macro F1-Score
DistilBERT	**Binary (Alerting)**	0.90	0.89
DistilBERT	Multiclass (Trends)	0.82	0.69
XGBoost	Binary (ML Champion)	0.57	0.73

🛠️ Technologies & Libraries

- **Primary NLP Framework:** Hugging Face `transformers` (DistilBERT)
- **Deep Learning:** PyTorch (`torch`)
- **Machine Learning:** Scikit-learn, XGBoost, `imblearn` (SMOTE)
- **Feature Engineering:** TF-IDF, GloVe Word Embeddings
- **Visualization & Deployment:** Streamlit, Matplotlib, Seaborn
- **Experiment Tracking:** Weights & Biases (`wandb`)

💻 How to Run the Project

The core project output is the interactive Streamlit dashboard.

1. Prerequisites

Ensure you have **Python 3.8+** and a package manager (`pip` or `conda`).

2. Clone the Repository

```
git clone git@github.com:FaithWambugu/NLP-Project.git
cd <repository_name>
```

3. Set Up the Environment

Install all required dependencies, including PyTorch and the Hugging Face libraries:

```
pip install -r requirements.txt
```

4. Launch the Streamlit Application

The application provides the final deployed solution, visualizing binary alerts and multiclass trends.

```
streamlit run sentiment_app.py
```

5. View the Full Analysis

All data cleaning, EDA, ML, and DL modeling steps are fully documented in the main Jupyter Notebook:

```
jupyter notebook
```

(Open the main notebook file and run the cells in sequence.)

🔍 Data Source & Structure

- **Source:** CrowdFlower Brands and Product Emotions dataset, available via `data.world` .
- **Total Records:** Over 9,000 tweets.
- **Key Columns:**
  - `tweet_text` : The raw text for analysis.

- `emotion_in_tweet_is_directed_at` : The specific brand or product.
- `is_there_an_emotion_directed_at_a_brand_or_product` : The original sentiment label (Positive, Negative, No emotion).

(Note: To complete the deployment, ensure your `requirements.txt` includes `torch` , `transformers` , `streamlit` , and `wandb` , and verify the paths to your saved DistilBERT models are correct in `sentiment_app.py`)