Name: Kamlesh Jadhav

Roll no: FT264071

News Headline Sentiment analysis

Project Report: Financial News Sentiment Analysis

a. Objective of the Exercise

The primary objective of this project is to perform and compare results of a Natural Language Processing (NLP) classification model RandomForest using two vectorization methods: BOW (CountVectorizer) and TFIDF vectorizer.

Primary objective:

1. **Data Ingestion:** Successfully load unstructured text data from a raw file source.
2. **Text Normalization:** Transform raw, noisy text into a structured format suitable for machine learning algorithms.
3. **Sentiment Classification:** Categorize headlines into three distinct classes: **Positive**, **Negative**, or **Neutral** to aid in automated financial trend analysis.
4. **Feature Engineering:** Convert textual data into numerical vectors using Bag of Words (BoW) and TF-IDF techniques to train predictive models.

b. Versions of Libraries Used

To ensure reproducibility and stability of the code, the following library versions were utilized in the Python 3.12.7 | packaged by Anaconda, Inc. | (main, Oct 4 2024, 13:17:27) [MSC v.1929 64 bit (AMD64)]

- **Pandas:** 2.2.2 (Data manipulation and analysis)
- **NumPy:** 1.26.4 (Numerical computing)
- **NLTK (Natural Language Toolkit):** 3.9.1 (Stopword removal and tokenization)
- **BeautifulSoup (bs4):** 4.12.3 (HTML parsing and tag removal)
- **SpaCy:** 3.8.11 (Advanced Lemmatization and linguistics)
- **Scikit-Learn:** 1.5.1 (Vectorization and Random Forest Modeling)

c. Acknowledgment of the Source of Data

- **Primary Source:** Kaggle.
- **Dataset Name:** Sentiment Analysis for Financial News
- **Origin:** This dataset contains sentences from financial news datasets, originally annotated by Malo, P., Sinha, A., Korhonen, P., Wallenius, J., and Takala, P. in their 2014 paper *"Good debt or bad debt: Detecting semantic orientations in economic texts."*

d. Meta Information about the Data

- **Dataset Structure:** The dataset consists of 4,846 rows and 2 columns.
- **Columns:**
    1. Sentiment: The target variable (Categorical: Neutral, Positive, Negative).
    2. Headline: The feature variable (Text data containing financial news).
- **Domain:** Financial and Economic news (focusing on earnings, stock moves, and corporate announcements).
- **Language:** English.
- **Class Distribution (Observed):**
    - The dataset is heavily skewed towards **Neutral** statements (common in financial reporting where facts are stated without emotion).
    - **Positive** and **Negative** sentiments are present but less frequent than Neutral.

e. Challenges Faced with the Data

During the data loading and cleaning phase, several technical and linguistic challenges were encountered:

1. **File Format Mismatch:** The file was named all-data.xlsx but the internal structure was actually comma-separated values (CSV), causing initial load failures with Excel parsers.
2. **Missing Headers:** The raw file lacked column headers, treating the first news headline as a header row, which required manual schema definition.
3. **Encoding Issues:** The file contained non-standard characters (likely Latin-1/ISO-8859-1) which caused utf-8 decoding errors during ingestion.
4. **Noisy Text:** The headlines contained HTML tags (e.g., <b>,), URLs, and special characters used in financial reporting (e.g., %, $) that create noise for the model.

5. **Stopword Sensitivity:** Standard NLP stopword lists remove words like "not" or "no." In sentiment analysis, removing these flips the meaning (e.g., "not good" becomes "good"), which is a critical error.
6. **Acronym Ambiguity:** The text was heavy with financial acronyms (e.g., "EUR", "mn", "PCBs") which standard dictionaries do not recognize.

f. Strategies to Overcome the Challenges

The following strategies were implemented to resolve the issues and create a robust pipeline:

1. **Robust Data Loading:**
    - Implemented a try-except block to attempt reading the file as Excel by saving the csv file as excel file
    - Manually assigned column names ['Sentiment', 'Headline'] in excel after loading to ensure no data loss.
2. **Customized Cleaning Pipeline:**
    - **HTML & URL Removal:** Used BeautifulSoup and Regex to strip formatting tags and hyperlinks before processing text.
    - **Acronym Expansion:** Created a domain-specific dictionary to map financial terms (e.g., EUR $\rightarrow$ euro, USD $\rightarrow$ dollar) after lowercasing, preserving their meaning by redefining the acronyms in lower case to ensure all the acronyms are taken care of.
3. **Context-Aware Stopword Removal:**
    - Instead of using the default NLTK stopword list, we created a **custom whitelist**. We mathematically subtracted negation words (e.g., no, not, never, didn't) from the stopword set. This ensured that negative sentiments were preserved in the final vectors.
4. **Advanced Text Normalization (Lemmatization):**
    - Moved from simple Stemming (which chops words blindly) to **Lemmatization using SpaCy**. This converted words to their root form based on context (e.g., "better" $\rightarrow$ "good", "running" $\rightarrow$ "run"), reducing the dimensionality of the vector space without losing semantic meaning.

g. Comparison of the two methods and their analysis metrics

Metric	Model A: Bag of Words (BoW)	Model B: TF-IDF
Overall Accuracy	75.15%	**75.57%**
Negative (0) Precision	0.72	**0.75**
Negative (0) Recall	**0.45**	0.43
Positive (1) Precision	0.76	**0.77**
Positive (1) Recall	**0.5**	0.48
Neutral (2) Recall	0.94	**0.96**

Observations:

A.Overall Performance (Accuracy) The two models perform very similarly, with **TF-IDF edging out BoW slightly (75.6% vs 75.2%)**.

Both models performed almost identically with ~73% accuracy. This indicates that for this specific dataset, the frequency of words (BOW) is nearly as predictive as the weighted importance of words (TF-IDF).

TF-IDF (Term Frequency-Inverse Document Frequency) assigns weights to words based on their uniqueness. In financial news, common words like "company", "said", or "market" appear in almost every headline regardless of sentiment. TF-IDF down weights these generic terms and gives higher importance to specific, sentiment-rich words (like "plummeted", "surged", "loss"), which helps the model make slightly cleaner distinctions.

B. The "Neutral" Class Dominance Both models perform exceptionally well on the **Neutral (2)** class, with Recall scores of 94-96%.

- **Observation:** The models correctly identify almost all neutral headlines but struggle significantly with Positive and Negative ones (Recall ~43-50%).
- **Reason:** This is a classic **Class Imbalance** issue. Your test set has 571 Neutral examples but only 110 Negative ones. The models have learned that predicting "Neutral" is the safest bet to maximize overall accuracy.

C. Precision vs. Recall Trade-off

- **TF-IDF has better Precision for Negatives (0.75 vs 0.72):** When the TF-IDF model predicts a headline is negative, it is more likely to be correct. This is because it focuses on rare, high-impact negative words.
- **BoW has better Recall for Negatives/Positives:** The Bag of Words model catches slightly more of the emotional headlines (higher recall), but at the cost of making more false alarms (lower precision).

h.Conclusion

Model B (TF-IDF) is the preferred model for this use case. although the accuracy gain is small, the **higher precision** is valuable in a financial context as we typically want to avoid false alarms (e.g., falsely tagging a neutral earnings report as "Negative" panic news). To improve performance further, we would need to address the class imbalance (e.g., using SMOTE or class weights) rather than just changing the vectorizer.

Repository link:

https://github.com/kamleshft264071-cpu/NLP-Take-home-assignment.git