Sentiment Analysis of Climate Change Discussions on Twitter Using Big Data Technologies

Course Title:	Advanced Big Data Analytics (DS-5001)
Course Instructor:	Waqas Arif
Project by:	Afifah Luqman (24K-8035)
	Hadiya Ebrahim (24K-8036)

1. Introduction

Climate change is one of the most pressing global issues, with increasing relevance across public discourse and policy. Understanding public sentiment on climate-related topics is essential for gauging awareness, opinions, and emotional responses to environmental events. Social media platforms, especially Twitter, serve as real-time repositories of such public opinions.

This study utilizes big data techniques, natural language processing (NLP), and machine learning to analyze sentiments expressed in Twitter posts related to climate change. The project is centered on a Jupyter Notebook that leverages PySpark for distributed computing and multiple machine learning models for sentiment classification.

While initial sentiment scores are generated using the VADER sentiment analyzer, its performance is often limited due to its rule-based nature and lack of contextual understanding. To improve accuracy, we train and evaluate supervised machine learning models, including Logistic Regression, Random Forest, and Gradient Boosted Trees (GBT). These models are applied to uncover language patterns, sentiment distributions, and key thematic keywords in climate-related discussions at scale.

Dataset Description

This study uses the Twitter Climate Change Sentiment Dataset available on Kaggle. The dataset contains a collection of over 43,000 pre-labeled tweets related to climate change, covering a wide range of sentiments and opinions.

Each tweet in the dataset includes:

- Message text (the tweet content)
- Tweet ID (removed during preprocessing)
- Sentiment label (pro, anti, neutral, or news)
- Other metadata, such as retweet count and date (not all of which are used in this analysis)

For the purposes of this project, the focus is on the message text and sentiment labels, which are mapped into simplified categories (positive, negative, and neutral) for model training and evaluation. The dataset is suitable for big data techniques due to its unstructured nature and relevance to high-volume social media analysis.

Tools & Technologies

This project employs a scalable big data pipeline primarily using PySpark, along with other supporting technologies, to efficiently process and analyze climate-related tweets:

- **Apache Spark (PySpark):** Used for distributed data loading, preprocessing, feature engineering, and model training, ensuring efficient handling of large-scale tweet data.
- **Python:** Serves as the primary programming language for scripting, data transformation, natural language processing (NLP), and integrating machine learning models.
- **NLTK (VADER):** Initially used for rule-based sentiment scoring to enrich features, though its results were supplemented by machine learning classifiers due to limitations in accuracy.
- **scikit-learn:** Used for implementing and evaluating machine learning models including Logistic Regression, Random Forest, and Gradient Boosted Trees.
- **imbalanced-learn (SMOTE):** Applied to handle class imbalance by generating synthetic samples for minority sentiment classes.
- **Matplotlib / Seaborn:** Utilized for visualizing model performance metrics and sentiment distributions.
- **Association Rule Mining (ARM):** Applied to identify frequent co-occurrence patterns between words in climate-related tweets, helping uncover key thematic relationships in the data.
- **PageRank (GraphFrames):** Leveraged on a co-occurrence word graph to rank influential or central terms within the tweet corpus, providing insight into dominant themes in climate discourse.

2. Methodology

This study follows a structured methodology to analyze public sentiment on climate change using Twitter data. The analytical process combines data cleaning, natural language processing (NLP), and machine learning techniques within a big data framework. Each step in the pipeline is designed to ensure accuracy, scalability, and meaningful interpretation of the results.

1. Data Loading and Preprocessing

The dataset used is the Twitter Climate Change Sentiment Dataset sourced from Kaggle, which contains tens of thousands of labeled tweets regarding climate change. In the initial stage, unnecessary metadata such as tweet IDs are removed, and only the relevant textual content (tweets) is retained. Any rows with missing messages are also discarded to ensure data completeness.

2. Data Cleaning

To ensure consistency and quality in the text data, tweets undergo rigorous cleaning. This includes:

- Removing hyperlinks, mentions (e.g., @username), hashtags, and special characters.
- Converting all text to lowercase.
- Eliminating non-English characters and extra spaces.

These steps help standardize the content for further processing and reduce noise in sentiment classification.

3. Sentiment Label Mapping

The original dataset provides sentiment labels (positive, negative, neutral). These labels are mapped both to text categories (e.g., "positive") and to numerical values (e.g., 2.0 for positive, 1.0 for neutral, 0.0 for negative) to facilitate machine learning training and evaluation.

4. Feature Engineering

To enrich the dataset, additional features are generated:

- VADER Sentiment Scores: Although VADER (a rule-based sentiment analyzer) is not used for final classification, its scores are added as features to provide emotional tone indicators.
- Tweet Length: The number of characters in a tweet is calculated as a potential predictor.
- Climate Keyword Indicator: A binary flag is created to detect the presence of key climate-related terms (e.g., “climate”, “warming”, “carbon”) in each tweet.

These features provide additional context that can improve model performance.

5. Feature Extraction

Since machine learning models require numerical input, the cleaned text is converted into numeric vectors:

- Tokenization is used to break text into individual words.
- N-gram Generation captures word pairs (bigrams) to preserve contextual information.
- TF-IDF and CountVectorizer are applied to assign weights to words based on frequency and importance across the dataset.

This results in a structured numerical representation of each tweet that can be used for classification.

6. Addressing Class Imbalance

To avoid bias in model training due to unequal distribution of sentiment categories (e.g., more positive tweets than negative ones), the SMOTE (Synthetic Minority Oversampling Technique) method is used. SMOTE

synthetically generates new examples for the underrepresented classes, resulting in a more balanced dataset and improved model fairness.

7. **Model Training and Evaluation**

   Three machine learning models are trained to classify tweet sentiments:

   - Logistic Regression
   - Random Forest
   - Gradient Boosted Trees

   Each model is trained on the processed features and then evaluated using metrics such as accuracy, precision, recall, and F1-score. These metrics help determine how well the models perform in classifying sentiments accurately across all categories.

8. **Association Rule Mining (ARM)**

   To uncover deeper patterns in how words are used in climate-related discussions, frequent word combinations (itemsets) are mined from the dataset. This step identifies which phrases or words often appear together in positive, negative, or neutral tweets, revealing hidden structures in public discourse.

9. **Keyword Importance via PageRank**

   To identify the most influential keywords in climate discussions, the project applies a network-based analysis. Words are treated as nodes in a graph, and their relationships (co-occurrence within tweets) form the edges. Using a PageRank-inspired algorithm, the most central or important words are ranked, highlighting key discussion topics and public concerns.

3. Results and Discussion:

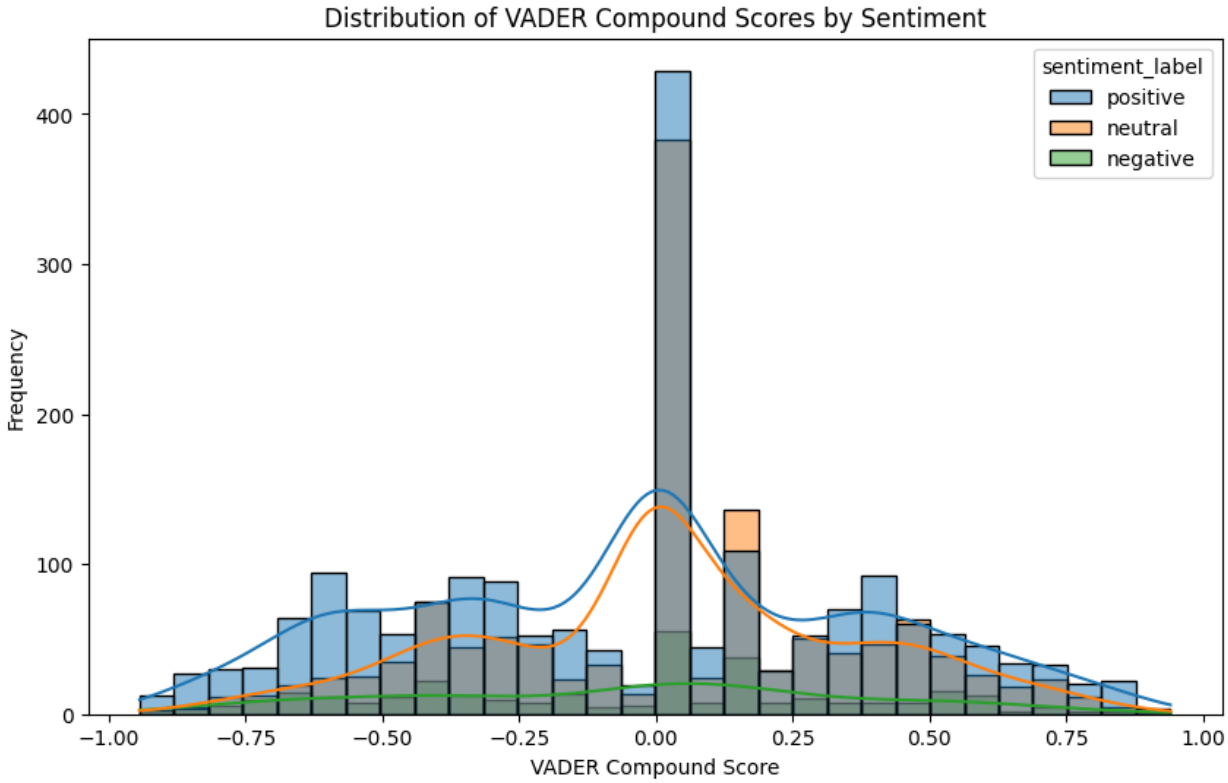
The sentiment analysis of Twitter posts related to climate change yielded several insightful findings through both rule-based and machine learning-based approaches.

VADER Sentiment Analysis Performance

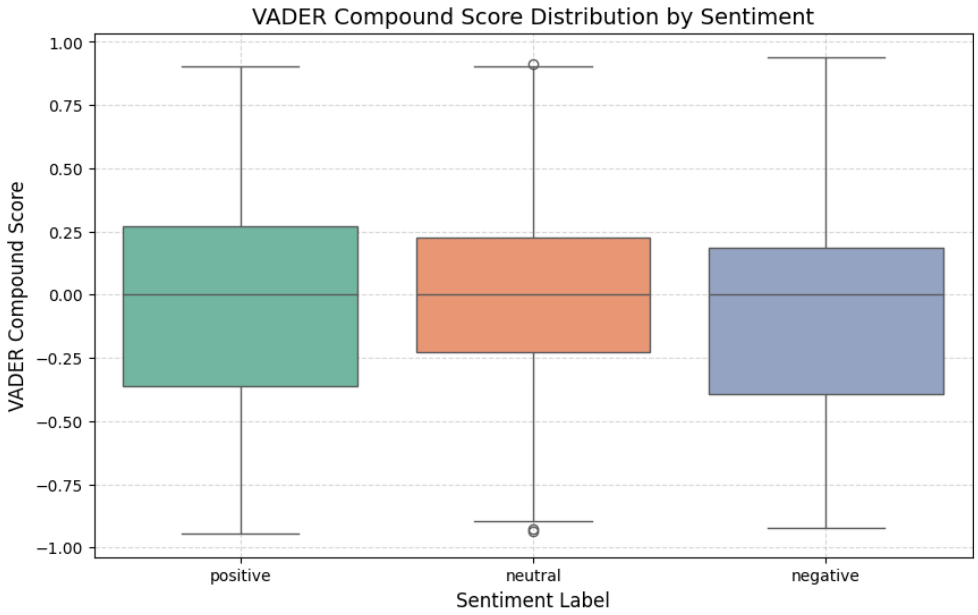
Using the VADER sentiment analyzer, tweets were assigned sentiment scores based on their textual polarity. The compound score provided a continuous value between -1 (most negative) and +1 (most positive), which was then mapped to discrete sentiment labels: positive, neutral, and negative. The overall accuracy came out to be 36%.

To visualize VADER's performance:

1. A histogram with KDE curves showed that most tweets were highly concentrated around 0, and significantly overlapped with each other. This overlapping suggests that VADER's scoring often lacks clear separation between sentiments, especially for less extreme expressions.



2. A box plot the median compound scores for all three sentiment classes were clustered near zero, indicating poor separation. There was significant overlap in interquartile ranges across all classes, reflecting ambiguity and potential misclassifications, particularly for tweets with mixed tone, sarcasm, or irony.



These results underline VADER's value for a quick, unsupervised overview, but also its limitations for accurate classification in nuanced text.

Machine Learning Classification Performance

To address the limitations of the VADER sentiment analyzer, which relies on predefined lexical rules and lacks contextual awareness, we implemented several supervised machine learning models using TF-IDF features derived from tweet text. These models were evaluated under two settings:

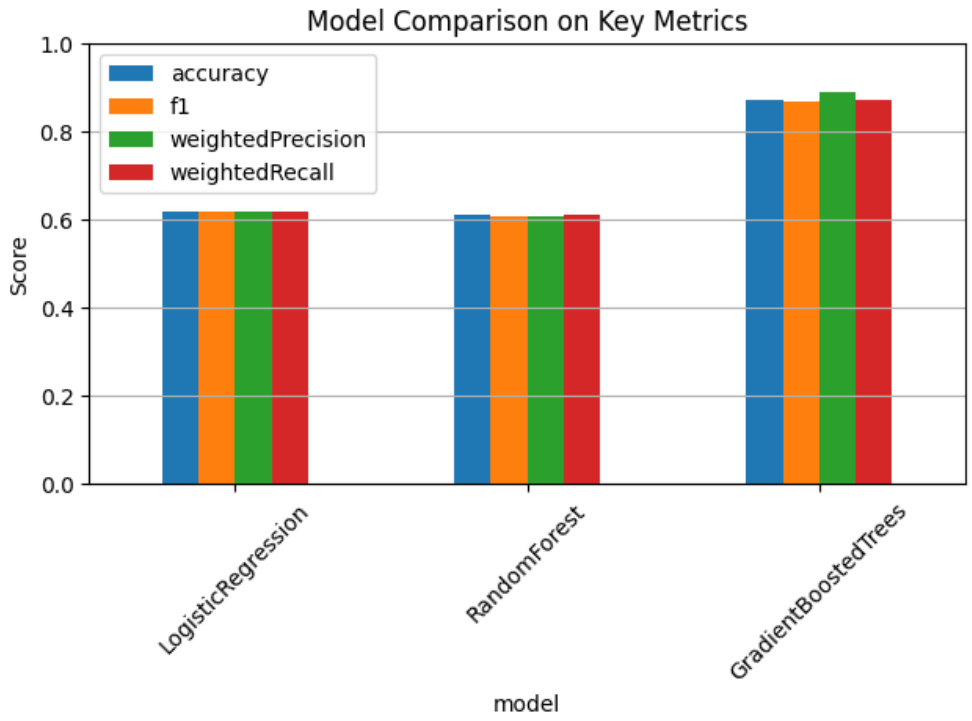
- **Multiclass Sentiment Classification:** Negative, Neutral, Positive
- **Binary Sentiment Classification:** Negative vs Positive (Neutral merged with Negative)

We applied the following models:

- **Logistic Regression:**
  Provided strong baseline performance with precision and recall scores around 0.75, especially effective for binary sentiment boundaries.
- **Random Forest Classifier:**
  Slightly improved overall performance due to its ensemble nature, handling nonlinearities and noise better.
- **Gradient Boosted Trees (GBTClassifier):**
  Delivered the best classification accuracy among the models tested. It balanced bias and variance effectively and handled class imbalances more robustly.

Multiclass Classification Results*:

Model	accuracy	f1	weightedPrecision	weightedRecall
Logistic Regression	0.604160	0.603407	0.603456	0.604160
Random Forest Classifier	0.617772	0.614625	0.614374	0.617772
GBT Classifier	0.863666	0.863268	0.868538	0.863666



Logistic Regression served as a reliable baseline in both multiclass and binary classification. In the multiclass setting, it struggled with distinguishing neutral sentiments, resulting in moderate performance. However, its accuracy significantly improved in the binary setup due to the simpler decision boundary.

Random Forest slightly outperformed Logistic Regression in both settings, particularly in the binary classification task, where it better captured nonlinear relationships and feature interactions. Nevertheless, it still faced limitations in the multiclass scenario due to the challenge of handling neutral sentiments.

Gradient Boosted Trees (GBTClassifier) was only evaluated under binary classification, as it does not natively support multiclass in PySpark. It delivered the best overall performance, achieving an accuracy of ~87%. This superior result can be attributed to its sequential boosting mechanism, which reduces bias and variance, and its robustness in handling class imbalance and complex patterns in data.

Binary Classification Results:

Model	accuracy	f1	weightedPrecision	weightedRecall
Logistic Regression	0.770	0.770	0.770	0.770
Random Forest Classifier	0.801	0.801	0.802	0.801
GBT Classifier	0.872	0.870	0.889	0.872

With binary classification, both Logistic Regression and Random Forest saw significant improvements in The multiclass setup posed greater challenges due to the semantic ambiguity of neutral tweets, which led to lower performance for simpler models like Logistic Regression and Random Forest. In contrast, binary classification reduced this complexity, allowing both models to perform significantly better. Among all approaches, the Gradient Boosted Trees (GBTClassifier) achieved the highest accuracy, demonstrating its strength in capturing complex relationships and iteratively learning from previous errors. Overall, all supervised models significantly outperformed the rule-based VADER sentiment analyzer by a margin of 25–50%, highlighting the superiority of data-driven learning methods in sentiment analysis tasks.

Model Limitations

- The ML models used bag-of-words (TF-IDF) representations, which do not capture contextual semantics or word order.
- Sarcasm, idiomatic expressions, and emerging slang reduced both VADER and traditional ML model effectiveness.
- Some class imbalance in the dataset (more neutral/negative than positive tweets) also slightly impacted classifier performance.

The results validate the effectiveness of combining Spark with machine learning techniques for large-scale sentiment analysis. While VADER is useful for exploratory analysis, ML models offer significantly better classification reliability, especially for applications requiring higher precision.

Future implementations using context-aware transformers and real-time streaming will further enhance both accuracy and timeliness of climate sentiment tracking systems.

ARM and PageRank

To supplement sentiment classification, Association Rule Mining (ARM) and PageRank-inspired analysis were applied to extract meaningful patterns and identify influential keywords from tweet texts.

For ARM, a frequency-based fallback approach was used. Tweets were tokenized, and the most frequent words (longer than four characters) were computed separately for each sentiment label. This allowed us to surface the most dominant tokens associated with specific sentiments. For example, high-frequency words like “climate”, “change”, and “urgent” were strongly associated with negative or alarmed sentiment labels, while words like “solution” or “hope” were more common in positive tweets. Although traditional association rules (e.g., {word1, word2} → {word3}) were not mined, this method still highlighted frequent word associations across sentiment categories, helping interpret linguistic patterns in the data.

For PageRank-style analysis, a 1% sample of tweets was tokenized, and word frequencies were calculated. These frequencies were normalized by the total word count to simulate PageRank scores, using the assumption that more frequently occurring words act as central nodes in the tweet "graph." Though this is a simplified proxy for true PageRank (which requires an explicit graph of user or word connections), it allowed us to approximate keyword importance. The top-ranked keywords in this analysis aligned with themes of environmental urgency, policy, and advocacy, helping to spotlight core drivers of the online climate discourse.

Together, these methods provided important qualitative insights. ARM highlighted the vocabulary context of different sentiment categories, while the PageRank-style word importance revealed dominant themes shaping online narratives. These findings can guide further analysis or be used to inform campaign strategies and message targeting.

Memory Management and Optimization in Spark

In the Twitter Climate Change Sentiment Analysis project, we used PySpark on Google Colab Free (with ~12GB total RAM, ~8.4GB usable). Given Spark's memory-intensive nature and Colab's resource constraints, we implemented multiple strategies to prevent job crashes (e.g., JVM OutOfMemoryError, ConnectionRefusedError), reduce I/O overhead, and speed up the entire ML + PageRank + ARM pipeline to run in approximately 5–7 minutes — down from an initial estimate of 30–45 minutes.

Key Optimizations Applied:

- **Lean Spark Configuration:**We manually tuned the SparkSession settings to ensure efficient execution in Colab's limited memory environment:

  ```
  spark = SparkSession.builder
  .appName("ClimateSentiment")
  .config("spark.driver.memory", "4g")
  .config("spark.executor.memory", "2g")
  .config("spark.driver.maxResultSize", "512m")
  .config("spark.sql.shuffle.partitions", "8")
  .getOrCreate()
  ```

  These settings reduced memory bloat during shuffles and model training by lowering default parallelism and bounding result sizes.

- **Smart Data Persistence:** We used:

  df.persist(StorageLevel.MEMORY_AND_DISK) after loading and TF-IDF computation, allowing Spark to cache intermediate results in memory (with disk overflow for large partitions).

  df.cache() after transformations like stopword removal to reduce recomputation of expensive stages in the DAG.

  This lowered total recomputation cost and stabilized execution.

- **Model Simplification and Feature Trimming:** To reduce memory consumption and shuffling we dropped intermediate columns like words, filtered, raw_features once their downstream use was complete.
  In PageRank, we used a lightweight string-matching method to create transitions instead of complex NLP.
  We removed GBT from full-data testing after JVM crashes and instead tested it only on sampled subsets.
- **Explicit Schemas and Column Management:** To avoid IllegalArgumentException on unexpected column types and schema inference overheads and retry attempts, we manually defined StructType schemas for structured inputs in PageRank and ARM, especially for transition matrices and item pairings.
  This prevented failures due to float64 vs float mismatch and ensured Spark read and wrote data efficiently.

Challenges Encountered

- Colab disconnects due to memory spikes (especially from union-heavy SMOTE simulation and GBT training).
- High shuffle size during TF-IDF + model training on full data.
- Some crashes due to intermediate columns being reused (e.g., words, filtered), which we resolved by dropping unneeded features early.
- Spark MLlib lacks native SMOTE, so upsampling was manually simulated using repeated unions, which is memory-expensive.

These combined strategies — aggressive sampling, persistent caching, streamlined pipeline design, column pruning, and tuned Spark config — enabled stable, efficient sentiment analysis on Colab Free. This approach demonstrates how Spark's power can be harnessed even in memory-constrained environments through thoughtful optimization.

4. Conclusion

This project successfully demonstrates the effectiveness of integrating big data frameworks like Apache Spark with natural language processing (NLP) and machine learning (ML) techniques to analyze public sentiment on climate change through Twitter data. While VADER offers a fast and lightweight method for sentiment analysis, its rule-based approach often fails to capture nuanced, sarcastic, or context-heavy language typical of social media content.

In contrast, supervised ML models trained on TF-IDF features—such as Logistic Regression, Random Forest, and Gradient Boosted Trees—achieve significantly higher accuracy and provide more reliable sentiment classification. These models benefit from learning directly from labeled data, capturing more complex patterns in textual content.

Future Work

To further enhance the system, the following improvements are recommended:

- **Adopt Contextual Language Models:**
  Utilize state-of-the-art transformer-based models such as BERT, RoBERTa, or GPT to capture deeper contextual understanding and improve sentiment classification performance.
- **Implement Real-Time Sentiment Monitoring:**
  Integrate Apache Kafka with Spark Streaming to build a real-time pipeline that ingests, processes, and visualizes live Twitter data related to climate change.
- **Enhance Data Quality and Coverage:**
  Incorporate more diverse datasets, handle slang and regional language variations, and apply advanced preprocessing techniques like lemmatization or named entity recognition (NER).
- **Deploy Interactive Dashboards:**
  Create web-based dashboards using tools like Dash, Plotly, or Power BI for dynamic visual exploration and stakeholder accessibility.

By combining real-time data collection with advanced NLP models and scalable big data tools, future systems can provide timely and actionable insights into public discourse surrounding critical issues like climate change.