Employee Sentiment Analysis Project: Technical Documentation

Date: July 4, 2025

Prepared by: Maulik Dave

---

1. Introduction

This document serves as the comprehensive technical documentation for the “Employee Sentiment Analysis” project. The initiative was launched with the intent to harness the power of natural language processing (NLP) and machine learning (ML) to analyze internal employee email communications. Through six modular tasks executed in Jupyter notebooks, the project constructs a full data pipeline beginning from raw text and culminating in a predictive model to estimate sentiment.

The core objectives include understanding the emotional tone of internal communication, identifying influential communicators, tracking sentiment trends over time, and building a foundational ML model to quantify sentiment based on textual and behavioral indicators.

---

2. Pipeline Summary and Code-Level Detail

Task 1: Text Preprocessing and Sentiment Analysis (task1.ipynb)

Objective: Convert raw email text into cleaned, lemmatized text and assign sentiment scores.

Techniques and Libraries:

- **NLTK** for tokenization, stopword removal, and lemmatization
- **VADER** sentiment analyzer for scoring text from -1 to 1

Output: task1_result.csv with columns: subject, body, from, date, processed_text, sentiment_score, and sentiment label (positive, neutral, negative)

Task 2: Sentiment Distribution Visualization (task2.ipynb)

Objective: Generate an initial visual overview of the sentiment scores across all messages.

Tools Used:

- **Matplotlib** to create a histogram of sentiment score frequencies

Insights Provided:

- Visual depiction of polarity distribution
- Early indication of communication climate

Task 3: Monthly Sentiment Aggregation (task3.ipynb)

Objective: Aggregate sentiment scores by sender and by month to analyze trends.

Techniques:

- Date parsing and formatting using pandas.to_datetime
- Grouping using groupby(['from', 'month'])

Output: task3_result.csv with from, month, and avg_sentiment_score

Task 4: Employee Ranking by Sentiment (task4.ipynb)

Objective: Identify the most positive and negative communicators monthly.

Method:

- For each month, extract the top 3 and bottom 3 employees by avg_sentiment_score

Output: task4_result.csv, a monthly leaderboard for employee sentiment

Task 5: Feature Engineering (task5.ipynb)

Objective: Prepare structured features for machine learning

New Features Created:

- message_length: Character count of body
- word_count: Word count of body

Output: task5_result.csv, the enriched base for modeling

Task 6: Predictive Modeling (task6.ipynb)

Objective: Predict sentiment_score using regression techniques

Features Used:

- message_length
- average_message_length
- message_frequency
- negative_word_count (via NLTK + opinion_lexicon)
- positive_word_count

Model 1: Linear Regression

- **MSE:** 0.2979
- **R-squared:** 0.1248

Model 2: Polynomial Regression (Degree 2)

- **MSE:** 0.2878
- **R-squared:** 0.1546

Conclusion: The linear model offers limited predictive capability; however, polynomial transformation offers mild improvement. Non-linear models are recommended for future iterations.

---

3. Correlation Matrix and Feature Insights

Feature Correlation with

Sentiment score

Feature	**Correlation (r)**
Positive word count	**0.2816**
Average message length	0.1573
Negative word count	-0.0842
Message frequency	0.0039

Key Observations:

- positive_word_count has the strongest positive correlation, suggesting that a higher density of positive sentiment words improves the sentiment score.
- average_message_length is weakly correlated but may interact with other features.
- message_frequency has negligible correlation and could be excluded in future versions.

---

4. Technical Evaluation

Linear Model:

- Weak fit, with R^2 ~0.12 indicating low explained variance
- RMSE suggests predictions deviate by over 50% of the target range

Polynomial Regression:

- Slight improvement in R^2 (~15%) and MSE
- Still linear in core nature; highlights the need for non-linear approaches like SVR, XGBoost

---

5. Future Recommendations

1. **Non-linear Modeling:** Explore tree-based models (Random Forest, XGBoost) or SVR with RBF kernel
2. **Textual Embeddings:** Replace manual counts with TF-IDF, BERT embeddings, or sentiment lexicon vectors
3. **Topic Modeling:** Integrate LDA for topic-driven sentiment clustering
4. **User Behavior Analytics:** Incorporate historical employee sentiment to track temporal trends

---

6. Execution Notes

To reproduce this analysis:

1. Place enron-spam-master.csv in the working directory
2. Install dependencies: pip install jupyter pandas nltk matplotlib scikit-learn
3. Run all notebooks sequentially from task1.ipynb to task6.ipynb
4. Ensure NLTK corpora are downloaded for sentiment and lexical analysis

---

7. Appendices

Appendix A: Library Versions

- Python 3.x
- Pandas 1.x
- NLTK 3.x
- Scikit-learn 1.x
- Matplotlib 3.x

Appendix B: Data Files Generated

- task1_result.csv
- task3_result.csv
- task4_result.csv
- task5_result.csv

End of Documentation