**Final Report — Employee Sentiment Analysis**

**1. Introduction**

Objective: Analyze employee messages to detect sentiment, compute monthly sentiment scores per employee, rank employees, identify flight risks and build a predictive model.

**2. Dataset and Preprocessing**

- Dataset file: `data/test(in).csv`

- Preprocessing steps:

- Convert `date` to datetime with `pd.to\_datetime(..., errors='coerce')`

- Drop rows with missing `from`, `body`, or `date`

- Add `month\_year` column using `date.dt.to\_period('M')`

**3. Sentiment Labeling (Task 1)**

- Method: NLTK VADER `SentimentIntensityAnalyzer`

- Label rule:

- compound >= 0.05 → Positive

- compound <= -0.05 → Negative

- else → Neutral

- Output: `labeled.csv` with `sentiment` column

**4. Exploratory Data Analysis (Task 2)**

- Missing values: shown and rows with critical missing info removed.

- Sentiment distribution: chart (`visualization/sentiment\_distribution.png`)

- Monthly trends: chart (`visualization/monthly\_sentiment\_trends.png`)

- Top senders: bar chart (included in notebook)

Key findings:

- (Insert bullet points derived from your EDA: e.g. majority neutral, spikes in months X and Y, top senders)

**5. Monthly Sentiment Scoring (Task 3)**

- Each message scored: Positive=+1, Negative=-1, Neutral=0

- Aggregate by employee and `month\_year`: `monthly\_sentiment\_score = df.groupby(['from','month\_year'])['sentiment\_score'].sum()`

- Output: `outputs/monthly\_scores.csv`

- Example:

- alice@example.com, 2010-07, +2

**6. Employee Ranking (Task 4)**

- We provide a 2–2–2 ranking per month: Top 2 Positive (>0), Top 2 Neutral (==0), Top 2 Negative (<0).

- Sorting: within category sorted by score then alphabetical for ties.

- Example table for latest month: `monthly\_rankings\_2\_2\_2.csv`

- Discussion: ranking logic and use cases for HR.

**7. Flight Risk Identification (Task 5)**

- Rule: employee flagged if they sent ≥4 negative emails in any rolling 30-day window.

- Rolling 30 days is computed per-employee by checking each negative message and counting negatives in the preceding 29 days.

- Result: `outputs/flight\_risks\_records.csv` and printed list (empty if none).

- Discussion: interpret empty result if no employees flagged.

**8. Predictive Modeling (Task 6)**

- Features used:

- message\_count\_month

- avg\_message\_length

- Model: Linear Regression

- Train/test split: 80/20

- Metrics: MSE, R-squared — see `outputs/regression\_results.csv`

- Visualization: Actual vs Predicted (`visualization/regression\_scatter.png`)

- Interpretation: discuss R² and feature importance

## 10. Reproducibility & Usage

1. Create virtual environment and install requirements: `pip install -r requirements.txt`

2. Put `test(in).csv` in `data/` (if allowed)

3. Run the notebook or `python src/pipeline.py`

4. Generated outputs will be in `outputs/` and visualizations in `visualization/`

## 11. Assumptions & Limitations

- VADER is rule-based and may misclassify domain-specific email language

- Flight risk metric (4 negatives in 30 days) is coarse — consider additional HR signals

- Predictive model uses only message features; richer features would improve performance.

## 12. Recommendations

- Add more features: sentiment intensity counts, n-grams, topic modeling, time-of-day features

- Human-in-the-loop review for flagged employees

- Regular monitoring dashboard for HR