Take-Home Assignment: Multimodal Rating Analysis with NLP

1. Objective
You are provided with an **artificial dataset (**https://github.com/Banking-Analytics-Lab/MultimodalFusionRatings/blob/main/Data/Artificial_Data.xlsx**)** that contains:
• **Structured Financial/Market Features** (e.g., FundaIndxint, monthvwretd, monthsprtrn, etc.).
• **Categorical Rating Columns** (RATING_TYPE, Rating).
• **Unstructured Text** in a column named string_values, containing short statements relevant to market conditions or company performance (e.g., *"Challenges remain in the supply chain, but strategic investments in logistics are improving efficiencies."*).

Your goal is to build a data science solution that leverages **both structured and textual data** to predict or explain the **Rating** (e.g., A+, BBB, etc.) or another related target (such as a numeric rating scale or future performance measure).
You will have 5–7 days to complete this assignment and must provide your final work in a **GitHub repository**.

2. Tasks
A. Data Exploration & Cleaning
1 Load & Inspect the Data
◦ Review column names, data types, and basic descriptive statistics.
◦ Identify missing values, outliers, or inconsistencies.
◦ Document any relevant observations (e.g., correlations, distributions).

2 Preprocessing
◦ Decide how to handle missing or inconsistent data (e.g., imputation, dropping rows/columns).
◦ Normalize or scale numeric features if necessary.
◦ Convert RATING_TYPE or Rating into suitable numerical or categorical encodings (if used as features or targets).

B. NLP Feature Engineering
1 Text Preprocessing
◦ Clean the string_values column: remove punctuation, lowercasing, stopword removal, and optional stemming or lemmatization.

2 Text Representation

◦ Choose a method (or multiple methods) to convert text into numeric features:

▪ **TF-IDF** vectors.

▪ **Word embeddings** (Word2Vec, GloVe).

▪ **Transformer-based embeddings** (e.g., BERT, Sentence Transformers).

3 Sentiment Analysis (Optional but Recommended)

◦ Use a sentiment analysis library (e.g., NLTK, TextBlob, Hugging Face) to generate sentiment scores and include these as features.

4 Topic Modeling (Optional)

◦ If there's enough data, run topic modeling (e.g., LDA) to uncover themes in the text. Incorporate topic distributions as additional features.

C. Predictive Modeling

1 Define Your Target

◦ **Classification**: Predict the categorical Rating directly (e.g., A+, BBB, etc.).

◦ **Regression**: Convert ratings to a numeric scale or select a numeric column (e.g., future return) as your target.

2 Model Development

◦ Train a **structured-only model** (no text features) as a baseline.

◦ Train a **text-only model** using your NLP features.

◦ Train a **combined model** that fuses both structured and text-derived features.

◦ Use any machine learning methods you find appropriate (e.g., Logistic Regression, Random Forest, Gradient Boosted Trees, Neural Networks).

◦ Perform hyperparameter tuning or cross-validation to improve performance.

3 Evaluation

◦ Use relevant metrics (e.g., Accuracy, F1-score, confusion matrix for classification; RMSE, MAE, R^2 for regression).

◦ Compare and report the performance of your three main approaches (structured-only, text-only, combined).

D. Interpretation & Insights

1 Feature Importance

◦ If using tree-based models, analyze feature importances or SHAP values.

◦ If using linear models, interpret coefficients or weights to see how each feature influences predictions.

2 NLP Insights

◦ Examine how sentiment or certain topics correlate with higher or lower ratings.
◦ Provide relevant visualizations (e.g., bar plots of average sentiment by rating, topic distributions).

3 Business Context

◦ Briefly discuss how your findings could inform real-world decision-making.
◦ For example, if negative sentiment in string_values correlates with rating downgrades, that might be a leading indicator.

3. Deliverables

All deliverables must be provided via a **GitHub repository**:

1 Code & Notebooks

◦ At least one Jupyter Notebook (or Python scripts) showing:
▪ Data loading, exploration, and cleaning.
▪ NLP preprocessing and feature engineering.
▪ Model training, evaluation, and comparison.
▪ Visualizations and interpretation.

2 README.md

◦ A concise explanation of:
▪ The problem and your approach.
▪ Steps to set up the environment (e.g., environment.yml or requirements.txt).
▪ How to run your code and replicate results.
▪ Key findings or highlights.

3 (Optional) Additional Documentation

◦ You may include a short PDF report or slides if you want to present your findings more formally.
◦ If you have multiple scripts, you can organize them into folders (e.g., src/,notebooks/).

4. Timeline

You have **5–7 days** to complete this assignment. This timeframe should allow for:
• Thorough data exploration and cleaning.
• Building and tuning at least one or two predictive models.
• Incorporating NLP features from string_values.

• Providing clear documentation in your GitHub repository.

5. Evaluation Criteria

1 Data Handling & Quality

◦ Thoroughness in dealing with missing data, outliers, and inconsistent formatting.

◦ Appropriate feature engineering for numeric and categorical columns.

2 NLP Integration

◦ Quality of text preprocessing, sentiment analysis, or embeddings.

◦ Effective demonstration of how textual features improve or complement structured data.

3 Modeling & Validation

◦ Logical approach to model selection and tuning.

◦ Clear, well-organized validation strategy (train/test split or cross-validation).

4 Interpretability & Insights

◦ Ability to highlight which features (numeric or textual) drive predictions.

◦ Relevant discussion of how sentiment or other text insights correlate with the target.

5 Code Organization & Clarity

◦ Readable, well-documented code.

◦ Clear instructions for reproducing your work (in the README).

Final Note

This assignment will showcase your ability to **blend traditional numeric data analysis with NLP** to generate deeper insights and improve model performance. We look forward to seeing your approach, creativity, and technical rigor in your final GitHub repository. Good luck!