**Agentic AI News Editor**

An autonomous system that selects, curates, rewrites, and explains news articles based on editorial goals. This project uses Retrieval-Augmented Generation (RAG) techniques and the Microsoft MIND dataset to create a complete end-to-end news curation system.

**Project Overview**

The Agentic AI News Editor is a minimal viable product (MVP) that demonstrates how AI can autonomously perform the following tasks:

1. **Retrieve and rank** news articles based on implicit editorial priorities
2. **Rewrite headlines** to maximize engagement while maintaining accuracy
3. **Explain editorial choices** in natural language
4. **Learn and improve** headline writing through continuous feedback

**Research Questions**

This project explores two main research questions:

**RQ1**: How effectively can the AI editor retrieve and rank news articles aligned with general editorial priorities and user needs?

* Metrics: Precision@5, category diversity score, user needs coverage, manual relevance check

**RQ2**: Does rewriting headlines via an LLM improve readability, clarity, and potential user engagement?

* Metrics: Flesch Reading Ease score, comparative headline evaluations, simulated click-through prediction

**System Architecture**

The system consists of the following components:

* **Data Connector**: Loads and processes news data from the MIND dataset
* **Headline Model Trainer**: Trains a model to predict headline click-through rates
* **Headline Metrics**: Evaluates headline quality based on various metrics
* **Headline Learning Loop**: Continuously improves headline rewriting through feedback
* **News Editor Controller**: Orchestrates all components
* **Web Frontend**: Visualizes the curated news in an interactive interface

**Getting Started**

**Prerequisites**

* Python 3.7 or higher
* OpenAI API key (for headline generation)
* Microsoft MIND dataset (small version recommended for testing)

**Installation**

1. Clone this repository:

git clone https://github.com/yourusername/agentic-news-editor.git

cd agentic-news-editor

1. Run the setup script:

python setup.py --full

1. Edit the .env file and add your OpenAI API key:

OPENAI\_API\_KEY=your\_openai\_api\_key\_here

**Running the System**

1. **Data Pipeline**: Process the MIND dataset and build the necessary indices:

python news\_editor\_controller.py --full

1. **Web Frontend**: Launch the Streamlit app:

streamlit run app\_frontpage.py

**Key Findings from EDA**

Our exploratory data analysis revealed several insights about headline effectiveness:

* Questions in headlines significantly reduce CTR (↓27.9%, p < 0.001) — best to avoid
* Numbers in headlines slightly reduce CTR (↓9.8%, p < 0.05) — use sparingly and only when truly meaningful
* Flesch Reading Ease Score shows no meaningful correlation with CTR
* News categories have a strong effect on CTR:
  + High CTR: kids, music, tv
  + Low CTR: autos, travel, northamerica

**System Components in Detail**

**1. Data Connector (data\_connector.py)**

Handles loading, processing, and encoding news articles from the MIND dataset:

* Loads news and behaviors data
* Processes raw data into a cleaner format
* Computes article embeddings using SentenceTransformer
* Builds a FAISS index for fast similarity search
* Calculates historical CTR for articles based on user behaviors

**2. Headline Model Trainer (headline\_model\_trainer.py)**

Trains a model to predict headline click-through rates:

* Extracts features from headlines
* Trains a RandomForest model to predict CTR
* Evaluates model performance
* Identifies important features for headline effectiveness

**3. Headline Metrics (headline\_metrics.py)**

Evaluates headline quality based on various metrics:

* Calculates predicted CTR using ML model
* Compares original and rewritten headlines
* Identifies key improvements
* Provides headline feedback

**4. Headline Learning Loop (headline\_learning.py)**

Continuously improves headline rewriting through feedback:

* Collects pairs of original and rewritten headlines
* Analyzes performance metrics
* Periodically retrains the CTR prediction model
* Generates insights about effective headlines

**5. News Editor Controller (news\_editor\_controller.py)**

Orchestrates all components:

* Prepares data
* Trains headline model
* Runs headline learning cycle
* Manages evaluation

**6. Web Frontend (app\_frontpage.py)**

Visualizes the curated news in an interactive interface:

* Displays curated articles
* Shows original vs. rewritten headlines
* Provides statistics on headline improvements

**Evaluation**

The system is evaluated based on:

1. **Retrieval Effectiveness**: How well it selects relevant articles
   * Topic coverage and diversity
   * Content relevance
2. **Headline Improvement**: How much it improves headlines
   * CTR improvement metrics
   * Readability changes
   * Structure analysis

Reports are generated automatically and saved to the evaluation\_results directory.

**Automating the Feedback Learning Loop**

The system includes a continuous learning mechanism:

1. New headline pairs are collected during curation
2. Performance metrics are analyzed
3. The model is periodically retrained
4. Insights are generated to guide future headline writing

**License**

This project is licensed under the MIT License - see the LICENSE file for details.

**Acknowledgements**

* Microsoft MIND Dataset
* OpenAI API for headline generation
* FAISS for efficient similarity search
* SentenceTransformer for article embeddings

**Authors**

* Your Name - Initial work

**Future Work**

* Implement A/B testing for headline variations
* Add personalization based on user preferences
* Expand to more diverse news sources
* Incorporate image selection and generation