Machine Learning Canvas – Online News Popularity

Designed for: Predicting the online popularity of news articles before publication.

Designed by: Team #41 - MLOps – Master's in Applied Artificial Intelligence, Tecnológico de Monterrey

Date: October 4th, 2025

Iteration: 1

PREDICTION TASK

- **Type of task**: Supervised learning (regression and/or classification).
- **Entity**: A single online article published on Mashable.
- **Possible outcomes**:
  - Regression: Predict the exact number of shares across social networks.
  - Classification: Predict popular vs. non-popular (using threshold = 1400 shares, as defined in Fernandes et al. 2015).
- **When outcomes are observed**: After publication, once social media share counts are collected from Facebook, Twitter, Google+, LinkedIn, StumbleUpon, and Pinterest.

DECISIONS

- **Editorial decision-making:**
  - Approve or modify articles before publication.
  - Adjust controllable factors (keywords, title sentiment, number of images, publication day).
- **Marketing strategy:**
  - Prioritize which articles deserve promotional budget.
  - Schedule content release for maximum impact (e.g., day-of-week effects).
- **Platform management:**
  - Select which articles to highlight on homepage and newsletters.
- The system integrates as recommendations inside the editorial workflow (CMS dashboard), providing real-time actionable predictions during article preparation.

VALUE PROPOSITION

- **Beneficiaries**:
  - Editorial teams: gain data-driven feedback to increase reader engagement.
  - Marketing teams: allocate promotional budget more efficiently.
  - Platform managers: boost site traffic, CTR, and ad revenue by prioritizing popular content.
- **Pain points addressed:**
  - Uncertainty about audience response before publishing.
  - High risk of investing in low-impact content.
  - Lack of predictive insight to optimize articles (titles, keywords, images).
- **Integration & workflow:**
  - A predictive service embedded into the content management system (CMS).
  - Predictions exposed via API (FastAPI service), easily accessible for editors.
  - Transparent recommendations (e.g., "Add more images", "Adjust title sentiment").
- **Business value:**
  - Reduce wasted promotion costs.
  - Maximize engagement and traffic.
  - Enable continuous learning: the system improves as more articles are published.
- **Educational relevance (for our project):**
  - This solution allows us to practice the entire MLOps lifecycle: data preparation, model building, experiment tracking, deployment, monitoring drift, and retraining.
  - By documenting this process, we not only solve the dataset challenge but also simulate how a real company would operationalize machine learning.

DATA COLLECTION

- **Initial dataset**: Online News Popularity dataset from UCI (2013–2015 Mashable articles).
- **Modified dataset**: Provided by course TAs, containing intentional noise and inconsistencies to test our skills in EDA, data cleaning, and preparation.
- **Strategy**:
  - Compare original vs. modified dataset to evaluate cleaning quality.
  - Apply systematic transformations: outlier removal, imputations, consistency checks.
- **Future updates (simulated):**
  - New article samples could be appended to simulate "streaming" data.
  - Costs controlled by using incremental updates and version control with DVC.

DATA SOURCES

- **Internal dataset (course-provided)**: Modified dataset with added random noise.
- **Original dataset:** Online News Popularity from UCI ML Repository.
- **External APIs (conceptual, future)**:
  - Social media APIs to track real-time shares.
  - Text analytics APIs and libraries (Pattern, spaCy, scikit-learn) for new feature extraction.
- **Experiment tracking**: GitHub repository for code, DVC for dataset versions, and MLflow for experiment results.

IMPACT SIMULATION

- **Costs of incorrect predictions:**
  - False Positive (predicting high popularity for a low-performing article): wasted promotion budget, editorial resources misallocated.
  - False Negative (predicting low popularity for a successful article): missed opportunity, lost traffic and revenue.
- **Simulated pre-deployment impact:**
  - Historic Mashable dataset with actual share outcomes.
  - Re-sampling strategies to test robustness of predictions.
- **Deployment criteria:**
  - For regression: reduce RMSE vs. naive mean predictor.
  - For classification: achieve ROC-AUC ≥ 0.70.
- **Fairness constraints:**
  - Ensure model does not penalize specific content categories (e.g., tech vs lifestyle).
  - Maintain transparency: editors understand why a recommendation was made.

MAKING PREDICTIONS

- **Mode**: Near real-time (batch predictions triggered when a draft is created or updated).
- **Frequency**: Each new article draft.
- **Latency tolerance:** ≤ 2 seconds per prediction (sufficient for CMS workflow).
- **Resources**:
  - Dockerized FastAPI service.
  - Cloud deployment (AWS/GCP/Azure).
  - CPU-based inference (dataset is medium-scale, GPU not required).

BUILDING MODELS

- **Number of models:**
  - Baseline regression model.
  - Advanced classification model (Random Forest / Gradient Boosting).
  - Potential ensemble.
- **Update frequency:**
  - Retrain monthly with new samples.
  - Trigger retrain if drift is detected.
- **Resources:**
  - CPU-based training feasible (~40k samples).
  - Containerized training jobs (Docker).
- **Pipeline:**
  - Organized with Cookiecutter structure.
  - Experiment tracking with MLflow.
  - Dataset versioning with DVC.

FEATURES

- **Representations at prediction time:**
  - Metadata: day-of-week, weekend flag, publication channel.
  - Text features: length, sentiment, subjectivity, keyword statistics.
  - Multimedia: number of images and videos.
  - LDA topic distribution (five-topic probabilities).
  - Self-reference shares of linked articles.
- **Transformations applied:**
  - Normalization and log-transformation of skewed features (shares).
  - One-hot encoding for categorical attributes.
  - Feature selection to avoid redundancy in keyword-related metrics.
  - Continuous monitoring of data drift between training and incoming articles.

MONITORING

- **Model performance metrics:** Regression (RMSE, MAE, R^2), Classification (Accuracy, F1, ROC-AUC)
- **Business KPIs:** Engagement metrics (average shares, CTR, time-on-page), ROI of promoted content, Growth in organic traffic.
- **Review frequency:** Technical metrics (weekly), Business impact (quarterly)
- **Tools:**
  - Evidently AI for drift detection and monitoring.
  - MLflow dashboards for experiment tracking.
  - GitHub & DVC logs for reproducibility and traceability.