ML System Design for **Personalized Newsfeed**

Objective: build a system that surfaces relevant, engaging, and personalized content for each user.

**Clarifying Questions**:

1. **User Goals**: What specific user actions (e.g., likes, clicks, shares, dwell time) are we aiming to increase?
2. **Content Freshness**: How important is it to prioritize real-time content (e.g., breaking news)?
3. **Real-Time Constraints**: How frequently should recommendations update to reflect recent user behavior?

**Define the ML Task**:

* **Recommendation Task**: This is primarily a recommendation problem where the goal is to predict which content each user is most likely to engage with.
* **Classification/Ranking Task**: Frame it as a ranking problem where content is scored and ordered based on its relevance to the user.

Data

**Inputs**:

**1. User Data**

* **User Profile Information:**
  + **Demographics: Age, gender, location, language.**
  + **Interests: Explicitly stated interests or inferred topics of interest.**
  + **Account Details: Date joined, subscription level (if applicable), device types used.**
* **User Behavior and Interaction History:**
  + **Engagement Metrics: Clicks, likes, shares, comments, reactions.**
  + **Browsing and Reading Patterns:**
    - **Dwell Time: How long they spend on an article.**
    - **Scroll Depth: How much of the content they scroll through.**
    - **Session Duration: Length of time per session.**
  + **Interaction Recency: How recently the user interacted with the app, and their interaction frequency.**
  + **Historical Data: Articles read, topics, categories, authors previously engaged with.**
  + **Time of Engagement: Times of day or days of the week they’re most active.**
* **User Preferences:**
  + **Content Type Preferences: Preferences for certain types of media (articles, videos, images).**
  + **Source Preferences: Preferred publishers, authors, or content sources.**
  + **Topics of Interest: Inferred from historical engagement data, explicit topic selections.**
  + **Sentiment/Emotion Analysis: Potentially inferred from reactions or comments, e.g., preference for positive or negative news.**

**2. Content Data**

* **Content Metadata:**
  + **Category and Topic: High-level categories (e.g., sports, politics) and specific tags or keywords.**
  + **Content Type: Format of the content, such as article, video, slideshow, or infographic.**
  + **Length and Complexity: Word count, reading level, and media richness (multimedia vs. text-heavy).**
  + **Author Information: Author profile, popularity, and prior engagement with the user.**
  + **Source/Publisher Information: Publisher reputation, brand preferences, etc.**
  + **Popularity Metrics: General popularity, such as overall likes, shares, or trending status.**
  + **Publish Date and Time: How recently the content was published, used to gauge freshness.**
* **Content Embeddings:**
  + **Precomputed embeddings of articles or videos based on NLP techniques or topic modeling. These can capture semantic relationships among content and help in aligning with user preferences.**

**3. Contextual Data**

* **Temporal Context:**
  + **Time of Day: Users may prefer different types of content at different times (e.g., quick reads in the morning, in-depth in the evening).**
  + **Day of the Week: Users may engage with different topics on weekdays vs. weekends.**
  + **Seasonality or Events: Specific trends or topics may peak around events (e.g., holiday seasons, elections, sports events).**
* **Device and Platform Context:**
  + **Device Type: Desktop, mobile, tablet—content layout and type may vary based on device.**
  + **Operating System and Browser: To optimize user experience, especially if specific types of content render better on certain platforms.**
  + **App vs. Web: Interaction patterns may vary depending on whether the user is using the app or the web version.**
* **Location-Based Context:**
  + **Geolocation: City, state, or country-level location, which can influence regional preferences.**
  + **Weather Information: Weather conditions may impact content preferences (e.g., planning activities).**
* **User Session Context:**
  + **Session Duration and Engagement: Whether it's the user’s first session of the day, total engagement length, and activity level during the session.**
  + **Current Activity: Engagement with specific topics or sections within the app during the session.**

**4. External Data Sources (Optional but Valuable)**

* **Trending Topics:**
  + **Real-time trends from social media platforms or news aggregators, which could help surface timely and relevant content.**
* **Sentiment Analysis:**
  + **Overall sentiment trends on topics to avoid overexposing users to negative news if they show a preference for positive content.**
* **Global and Local Events:**
  + **Major events or crises that may influence what users want to see (e.g., elections, natural disasters).**

**Outputs**:

* A ranked list of content items that are personalized for each user’s feed.

**Model Selection**:

**1. Collaborative Filtering**

* **Pros: Learns from user-item interactions, capturing latent patterns in user preferences without needing content data.**
* **Cons: Suffers from the "cold start" problem for new users and new content. Limited if users don’t have a lot of interaction history.**
* **Implementation: Matrix factorization models like SVD or neural approaches like Neural Collaborative Filtering (NCF) can be used to predict user-item affinity. NCF can capture more complex patterns than traditional matrix factorization.**

**2. Content-Based Filtering**

* **Pros: Works well for new or niche content by analyzing the content itself. Can recommend relevant items even if the user is new, as long as we have their initial preferences.**
* **Cons: Limited personalization since it relies on the user’s profile and may over-recommend similar items, lacking diversity.**
* **Implementation: Content embeddings (e.g., BERT embeddings for text) are used to represent articles, and cosine similarity or other distance metrics can recommend items with similar features.**

**3. Hybrid Models**

* **Pros: Combines the strengths of collaborative and content-based filtering, improving personalization and handling both cold start (content-based part) and diversity (collaborative part).**
* **Cons: Higher complexity and requires careful tuning to balance both methods. Computationally more intensive.**
* **Implementation: Use collaborative filtering as the base, with content-based recommendations added when the user-item interaction is sparse. Methods like weighted hybrid or model-based hybrid combine predictions from both models.**

**4. Multi-Task Learning Models**

* **Pros: Can optimize for multiple signals simultaneously (e.g., clicks, shares, dwell time), potentially providing more holistic and engaging recommendations.**
* **Cons: More complex to train and requires labeled data for each task. Also, challenging to balance all signals effectively.**
* **Implementation: Use deep learning architectures with shared layers and separate heads for each task (e.g., clicks, shares, etc.), enabling the model to capture patterns that benefit multiple engagement metrics.**

**1. Hybrid Model Design**

A hybrid model can combine collaborative filtering and content-based filtering, often using a two-tower (dual-encoder) architecture:

**a. Collaborative Filtering Component:**

* **User Embedding**: Represent each user based on their interaction history (e.g., items liked, clicked, shared) using embeddings. This can be learned through matrix factorization or neural collaborative filtering (NCF).
* **Item Embedding**: Represent each item based on its interaction patterns with users. Items with similar interaction histories will have embeddings closer to each other in the latent space.

**b. Content-Based Filtering Component:**

* **Content Embedding**: Generate content embeddings for items based on text features (e.g., article title, summary, keywords) using pre-trained language models like BERT, which can capture contextual nuances.
* **Metadata Embeddings**: Create embeddings for metadata features such as category, author, or publish date. Concatenate these embeddings with content embeddings to enrich the content representation.

**c. Combined Architecture:**

* Use a **two-tower neural network** where:
  + One tower is dedicated to the **user embedding** (from collaborative filtering).
  + The other tower is for the **item embedding**, which is the combination of collaborative and content-based embeddings.
* **Dot Product or MLP**: Use a dot product or a Multi-Layer Perceptron (MLP) to combine the user and item embeddings, generating a relevance score for each user-item pair.
* **Training**: Train the model to maximize similarity for user-item pairs with positive interactions (e.g., clicked, liked) and minimize similarity for negative interactions (e.g., skipped or not clicked).
* **Cold Start Handling**: For users/items with no interaction data, rely only on the content-based embeddings. Gradually incorporate collaborative data as interactions build up.

**Output: The model outputs a relevance score for each item for a given user, ranking items based on their relevance.**

**2. Multi-Task Learning Model Design**

A multi-task learning model optimizes for multiple engagement signals (e.g., clicks, shares, dwell time), aiming to balance these objectives for more holistic recommendations.

**a. Shared Embedding Layers:**

* **User Embedding**: Create a shared user embedding layer based on interaction history, demographic data, and other personal attributes.
* **Item Embedding**: Create a shared item embedding layer from content features, collaborative features, and item metadata.
* Use these embeddings as inputs for multiple tasks.

**b. Task-Specific Heads:**

* Each task represents a different user engagement signal (e.g., clicks, shares, dwell time).
* Create separate task-specific output layers for each engagement metric, with each output layer designed to predict one engagement type.
* **Task Heads**:
  + **Click Prediction Head**: A binary classification head that predicts the likelihood of a user clicking on an item.
  + **Share Prediction Head**: Another binary classification head to predict if a user will share an item.
  + **Dwell Time Prediction Head**: A regression head to predict how long a user will spend reading the item.
* **Loss Functions**: Use different loss functions for each head (e.g., cross-entropy for clicks/shares, mean squared error for dwell time), with weights to balance each task’s importance.

**c. Training:**

* Train the model jointly on all tasks by minimizing a weighted sum of the individual losses for each task. This enables the shared embedding layers to learn representations that are useful across tasks, while each task-specific head fine-tunes for its respective signal.
* Adjust task weights based on business priorities or model performance on each task.

**Output: The model outputs separate predictions for each engagement metric, which can then be combined into an overall relevance score.**

**Comparison and Selection**

For a **personalized newsfeed**, if the goal is to maximize general engagement while addressing cold start and diverse content, a **Hybrid Model** is generally a good choice. It allows for robust user-item recommendations even with limited interaction data and adapts well to user interests.

If the business prioritizes multiple engagement signals simultaneously and has enough labeled data for each, a **Multi-Task Learning Model** can be more effective, providing a well-rounded view of user preferences across different actions.

**Final Selection: Hybrid Model with Multi-Task Learning Extension**

For best results, you can start with a **Hybrid Model** and, once it’s established, add task-specific heads to capture additional engagement signals through multi-task learning. This combines the strengths of both models, supporting personalization and multi-objective optimization for engagement.

Feature Engineering

**1. User Features**

* **Demographic Features**:
  + **Age Bucketing**: Group ages into ranges (e.g., 18-24, 25-34) to generalize and reduce dimensionality.
  + **Location Encoding**: Use one-hot encoding or regional embeddings for user location (city, state, or country).
  + **Language Preferences**: One-hot encode or use embeddings to capture preferred languages.
* **Behavioral and Interaction Features**:
  + **Click-Through Rate (CTR)**: Calculate CTR per topic, content type, or category. These features show which types of content the user tends to click on.
  + **Engagement Score**: Combine clicks, likes, shares, comments, and average reading time into a single engagement metric, weighted to emphasize certain actions.
  + **Recent Interactions**: Create features representing recent interactions, such as the topics of the last few articles the user clicked on, read, or engaged with.
  + **Time of Day and Day of Week Preferences**: Aggregate engagement data by hour and day, capturing when the user is most active, which can be useful for real-time recommendations.
  + **Content Consumption Diversity**: Measure the diversity of topics the user engages with (e.g., entropy of topic distribution). Users with high diversity may need more varied recommendations.
* **Historical Interest Trends**:
  + **Topic Affinity**: Track long-term interests by calculating average engagement metrics per topic (e.g., “sports,” “technology”) over a long history.
  + **Temporal Interest Decay**: Apply a time decay factor to engagement scores, giving more weight to recent activity than older activity. Exponential decay functions work well here.
* **Session-Based Features**:
  + **Session Length and Frequency**: Track the average number of articles consumed per session and the frequency of sessions per day or week.
  + **Scroll Depth and Completion Rate**: Features to capture whether users read articles fully, partially, or just skim.

**2. Content Features**

* **Text-Based Features**:
  + **Content Embeddings**: Use NLP models (e.g., BERT or word2vec) to create embeddings of article text, capturing the semantic meaning. This can be used to match content to similar articles.
  + **Topic Modeling**: Use techniques like LDA or clustering to assign topics or tags to articles, converting content into a more structured form.
  + **Sentiment Analysis**: Analyze the article text to derive sentiment scores. These scores can help match content to users who prefer positive or negative tones.
* **Metadata Features**:
  + **Category and Subcategory**: One-hot encode or use embeddings to represent the high-level category of the article (e.g., sports, tech, health).
  + **Length of Content**: Use article length as a feature. For example, short vs. long content can be important for matching user session length preferences.
  + **Author Popularity**: Track and add engagement metrics (e.g., average article views) for each author, so popular authors can be boosted in recommendations.
* **Popularity and Trending Scores**:
  + **Global Popularity**: Overall engagement metrics (e.g., clicks, shares) for each piece of content, indicating general popularity.
  + **Relative Popularity**: Calculate popularity within specific demographics or regions if the user base is diverse.
  + **Trend Score**: A time-weighted metric that emphasizes recent engagement, used for trending content.

**3. Contextual Features**

* **Time and Day**:
  + **Time of Day**: Include features for the current time, such as morning, afternoon, evening, and night (could be one-hot encoded).
  + **Day of the Week**: Encode the current day, as user engagement may differ across days (weekdays vs. weekends).
  + **Season or Event Indicator**: For special events or seasons (e.g., holiday season), use binary features to indicate if a specific period applies to the current time.
* **Device and Platform Features**:
  + **Device Type**: Encode whether the user is on mobile, desktop, or tablet.
  + **App vs. Web**: Capture the platform type to adjust the type of recommended content (app users may engage differently than web users).
* **Location Context**:
  + **Geolocation (City, Region)**: Use one-hot encoding or embeddings to capture the user’s location.
  + **Weather Features**: Add features like temperature or weather conditions if location data is available, as weather may influence content preferences.

**4. Feedback and Real-Time Features**

* **Recent Engagement Indicators**:
  + **Session-Based Engagement**: Capture engagement metrics for the current session (e.g., number of clicks so far, time spent).
  + **Previous Article Engagement**: For the latest articles the user interacted with, create features indicating topic, sentiment, and engagement to tailor the next recommendations.
* **Feedback Signals**:
  + **Explicit Feedback**: Features that capture explicit user feedback (e.g., ratings, thumbs up/down).
  + **Negative Interaction Signals**: Track actions like hiding posts or marking content as irrelevant to avoid recommending similar items.

**5. External Data Features**

* **Trending Topics (External Source)**: Include external trending data (e.g., from social media) to boost content related to trending topics in recommendations.
* **Sentiment Trends**: Use overall sentiment trends for a topic or region, which may help align with current user mood or preferences.

**Evaluation Metrics**

1. **Offline Metrics:**
   * **Precision, Recall, F1 Score: For testing accuracy in recommending relevant items.**
   * **NDCG (Normalized Discounted Cumulative Gain): To evaluate ranking quality based on user engagement.**
   * **Coverage: Ensuring diverse recommendations across categories.**
2. **Online Metrics:**
   * **CTR: The click-through rate on recommended items.**
   * **Dwell Time: Measures user engagement with recommended articles.**
   * **Conversion Rate: Clicks or engagements that lead to subscriptions or further actions.**
   * **Bounce Rate: Lowering the rate can indicate higher relevance.**

A screenshot of a computer

Description automatically generated

https://www.youtube.com/watch?v=JiGrpySlVA0&list=PLlvnxKilk3aKx0oFua-HTtFf-d\_inQ8Qn&index=9