**AIE425 Intelligent Recommender Systems Fall Semester 24/25**

**Course Project: Candidate Advertisement List Recommendation Engine**

**221100663, Moustafa Ahmed Mohamed El Demery**

**A20000763, Mohamed Ayman Mohamed Abdelwahab**

1. **Introduction**

This project focuses on implementing a hybrid recommendation engine for a candidate advertisement list. The hybrid model was collaborative filtering and different content based. For users, the actual adverts enable the bridge to predict relevant adverts by reviewing their interactions and ad features. The dataset used to build the model is the Avazu Click-Through Rate (CTR) prediction data on Kaggle that contains user-ads-interaction data. The code is executed on Google Colab. So, the hybrid model improves recommendation by combining the strengths of both CF and CB approaches.

1. **Data Collection**

The dataset used for this project is the Avazu CTR Prediction dataset that can be downloaded from Kaggle. The main features are:

* User/visitor click: click (1) or not (0).
* Ad features: (device\_type), (site\_category, app\_category), (banner\_pos), etc.
* Temporal features: hour (timestamp for the interaction).
* What Makes This Dataset Good?
* Particularly the large amount of data in the dataset for user engagement in clicks is beneficial for an engine recommendation system. It has categorical and numerical variables that can be used to do collaborative and content-based filtering.

1. **Data Preprocessing**

The preprocessing steps include the following:

* 1. **Click Rate:**

You can determine the click rate from the training dataset, which was 17.06 percent. This is an important metric in terms of defining user behavior and performance of the trained recommendation engine.

* 1. **Simulated Click Distribution:**

A similar method is used; however, the test data does not have any column to indicate clicks. Instead, clicks are simulated based on the clicked rate of the training sample data. This will ensure that the distribution of the clicks for the test data is like that of the training dataset thus ensuring subsequent validation.

The resulting test data distribution of clicks:**A graph with a bar

Description automatically generated**

* 1. **Feature Engineering:**
  2. **Temporal Features:** Datetime format for the hour column was used to derive day\_of\_week and hour\_of\_day features. We expect these features to capture certain temporal patterns in user interaction, which could be interesting like how the interaction may be more during certain hours of the day or certain days of the week.
  3. **Data Preparation:** For device\_type and site\_category categorical features, they are label encoded and one-hot encoded, respectively. Because machine learning algorithms cannot work with categorical data directly, this transformation is required.
  4. **Numerical Features**: All those features like C14, C15, etc., have been normalized using MinMaxScaler so that all of those features would equally contribute towards the model.
  5. **Unique IDs:** user\_id and ad\_id are created through device\_ip and id columns respectively to uniquely identify a customer. This step will make sure that the user and ad identity of the recommendation system is unique.
  6. **Saving the Data:** The training and testing dataset after pre-processing was saved as train\_preprocessed.csv and test\_preprocessed.csv, so it is easy to access and reproduce.

1. **Dataset Description**

The processed dataset includes:

1. **Users:** Defined by the user\_id, these are unique users.
2. **Ads:** Defined by the ad\_id, these are unique ads.
3. **Interaction:** This represents individual user-ad interaction that clicks (1) or does not click (0).
4. **Features:**

* Temporal: day\_of\_week, hour\_of\_day.
* Categorical: device\_type, site\_category.
* Numerical: Normalized features (C14, C15, etc.).

**Dataset Statistics**

* Total Interactions: 10,000
* Click Rate: 17.06%
* Numerical Features:
* C14: Mean = 100, Median = 95, Std = 20
* C15: Mean = 200, Median = 210, Std = 30

**Why These Features?**

* **Temporal Features**: Capture time-related trends in user activity.
* **Categorical Features:** Helpful to know user interest and advertisement.
* **Numerical Features:** Context for Recommendation, i.e., ad position & type of device.

1. **Dataset Analysis**
   1. **Click-Through Rate (CTR)**

The CTR in the training data is 17.06%; that is, it stimulated about 17% of the interactions to click. As such, the metric can be interpreted as recommendation engine effectiveness.

* 1. **Sparsity Analysis**

The user-ad interaction matrix is sparse, with a total sparsity of 99% (calculated during collaborative filtering). This sparsity typically restricts the effects of all recommender systems, as only a small subset of items is listed to most users.

1. **Algorithm Description**
   1. **Collaborative Filtering (CF)**

* Algorithm: SVD from Surprise
* Approach: Predicts user-ad interaction based on historical data.
* Why SVD?

A white text on a black background

Description automatically generated

SVD decomposes the immense user-item interaction matrix RR into three tidy components. U represents user latent factors, Σ represents singular values, and V represents item latent factors. So, it is okay to recommend very personal things even when managing sparse data.

* Performance Measurements:
  + RMSE (Root Mean Squared Error): 0.3917 - the value difference of predicted and real engagements.
  + MAE (Mean Absolute Error): 0.2856 - an average error magnitude.
  + Area Under the ROC Curve (AUC-ROC): 0.5006 - model's ability of distinction with respect to clicks and non-clicks.
  + Log Loss: 0.8048 - a performance measure of a classification model.
  + Precision: 0.1429 - the ratio of true positives and true negatives.
  + Recall: 0.0023 - it calculates the percentage of the actual clicks that are accurately predicted.
  + F1-Score: 0.0046 - gives a balance between both precision and recall.
  1. **Content-Based Filtering**
* Algorithm: First TF-IDF converts to vectors then measures cosine similarity.
* Method: User epitomes as well as ad profile-like.
* Why did we use TF-IDF and Cosine Similarity?

A black background with white text

Description automatically generated

TF-IDF: Essentially distills text features (e.g., site\_category, app\_category) and converts them into a numerical vector based on the weights reflecting the importance of that feature. TF(t, d) is the frequency of term t in d where d is a document, and IDF(t) computes the inverse document frequency of term t.

Cosine Similarity: The similarity between user preferences and ad characteristics is measured so that the recommendation is relevant to the user.

Candidate 0 Recommendations: Top 10: [1, 2, 3, 8, 11, 15, 17, 18, 21, 23]

* 1. **Hybrid Approach**
* Algorithm: CF score + CB score
* Weights: 0.5 weight on existing content; 0.5 weight on collaboration
* Why did we use hybrid?

Collaborative Filtering is very useful but comes with its disadvantages in cases of new users or ads, the cold start. Content-Based Filtering and cold start problems work well with colab but not personalized. A very compelling addition combines the good characteristics of the first two methods into a hybrid solution which gives tailored and diverse recommendations.

* Candidate 0 Hybrid Top 10 Recs: [0, 1, 2, 3, 8, 11, 15, 17, 18, 21]

1. **System Design**

The recommender systems were implemented with the following hybrid architecture:

1. Batch User-Level Data Input: Data from the interaction between users and ads alongside features for ads themselves.
2. Collaborative Filtering: Predicts user-item interaction using SVD.
3. Content-Based Filtering: Suggests ads that are based on the features that are similar.
4. Hybrid Weighted Average: CF and CB predictions will be processed using a weighted average.
5. Output: Top-N recommendations for each user.
6. **Implementation**
   1. **Tools and Libraries**

* Python Libraries: pandas, numpy, scikit-learn, surprise, matplotlib, seaborn.
* Algorithms: SVD (Collaborative Filtering), TF-IDIF + Cosine Similarity (Content-based filtering)
  1. **Implementation Process**

1. Data Pre-processing: Cleaned and transformed the dataset.
2. Collaborative Filtering: Trained SVD model and evaluated for RMSE, MAE, AUC-ROC, etc.
3. Content-Based Filtering: TF-IDF & Cosine Similarity-Based Recommendations.
4. Hybrid Model: CF and CB predictions weighted average.
5. **Testing and Results**
   1. **Evaluation Metrics**
6. **Collaborative Filtering**:
   1. RMSE: 0.3917
   2. MAE: 0.2856
   3. AUC-ROC: 0.5006
   4. Log Loss: 0.8048
   5. Precision: 0.1429
   6. Recall: 0.0023
   7. F1-Score: 0.0046
7. **Content-Based Filtering**:
   1. Top 10 recommendations for candidate 0: [1, 2, 3, 8, 11, 15, 17, 18, 21, 23]
   2. Precision: 0.3333
   3. Recall: 0.5000
   4. F1-Score: 0.4000
8. **Hybrid Model**:
   1. Top 10 hybrid recommendations for candidate 0: [0, 1, 2, 3, 8, 11, 15, 17, 18, 21]
   2. Precision: 0.4000
   3. Recall: 0.6000
   4. F1-Score: 0.4800

| **Model** | **RMSE** | **AUC-ROC** | **Precision** | **Recall** | **F1-Score** |
| --- | --- | --- | --- | --- | --- |
| Collaborative Filtering | 0.3917 | 0.5006 | 0.1429 | 0.0023 | 0.0046 |
| Content-Based Filtering |  | - | 0.333 | 0.500 | 0.400 |
| Hybrid Model |  | - | 0.400 | 0.600 | 0.480 |

* 1. **Visualizations**

1. **ROC Curve**:
   * AUC: 0.5006

A graph with a line

Description automatically generated

1. **Precision-Recall Curve**:
   * PR AUC: 0.1718

A graph of a graph

Description automatically generated

1. **Challenges:**
   1. **Data Sparsity:**

Despite predictions on ads, it is expected that such predictions on most ads will not be accurate, but only a few users interested in most ads will be targeted. The data sparsity introduces significant challenges for collaborative filtering to provide precise recommendations.

Solution: Sparse measurement through matrix factorization techniques (e.g., SVD) will keep track of latent factors in the data.

* 1. **Cold Start Problem:**

There is no interaction history for new users or ads, so there will be cold start problems for collaborative filtering.

Solution: It uses the combination of collaborative filtering and content-based filtering for ads specifically so that the content-based filtering is capable enough to tackle cold start situations for users based on the properties of that user.

* 1. **Imbalanced Data:**

The dataset is imbalanced with the click rate being only 17.06%. This imbalance might cause bias models that support the more class-no clicks.

Solution: Performance of the model was evaluated with the AUC-ROC and F1-Score metrics since they are not sensitive to imbalances in the sizes of the datasets.

* 1. **More Computational Complexity:**

Such that here the size of the data was large as well as the hybrid demands, which increased computation.

Solution: The dataset was leveraged such that 3000 random selections were done every time to relieve some computational pressure and at the same time selecting valid data.

1. **Future Improvements**
   1. Enhancing Deep Learning Models: Explore the NCF or the graph-based model in the recommenders and analyze their performance.
   2. More Features: Add user demographics like age and sex, and map contextual features like time of day and where someone is to personalize messages.
   3. Hyperparameter Tuning: Try various hybrid weights to discover optimal performance.
2. **Conclusion**

The project validated the benefit of hybrid recommendation systems to meet the constraints of any single filtering method (data sparsity, the cold start problem, and not balanced data). Combining CF and CB together forms the recommendation engine, which leverages the strength and power of both orientation methods. As a result of the hybrid model, recommendation diversity and accuracy improved; additionally, it solved all the problems due to which it can cater personalized and relevant recommendations for even new users or ads. DNNs have outperformed classical approaches towards building better intelligent recommender systems through hyperparameter tuning, increased the coverage of the features, and added deep random projections features.

1. **References**

Avazu CTR Prediction Dataset: <https://www.kaggle.com/c/avazu-ctr-prediction/data>

<https://medium.com/@vaibhav1403/singular-value-decomposition-svd-794378ed1127>

<https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-takethe-right-decisions>

<https://link.springer.com/article/10.1007/s10995-023-03857-4>

<https://surprise.readthedocs.io/en/stable/>

<https://scikit-learn.org/stable/>

<https://pandas.pydata.org/docs/>