Maximizing the Value of Your Content

How to Build an Effective Content Recommendation Engine

IN PARTNERSHIP WITH:

Google News Initiative
OVERVIEW

BUSINESS REQUIREMENTS
What Would a Publisher Need Prior to Getting Started?

CONTENT RECOMMENDATIONS
Data Collection
Architecture
Data Collection Steps
  [ Step 1 ] Log Export: Filter
    Filter
    Description
  [ Step 2 ] Pub/Sub Message: Dataset (GA View)
  [ Step 3 ] Cloud Function: List_article_urls
    SQL
  [ Step 4 ] Cloud Function: Analyze_content
  [ Step 6 ] Google BigQuery: nlp_analysis
    Big Query Datasets
    BigQuery Table: Articles
    BigQuery Table: nlp_results
User Interests Scores
  Creating User Interest Scores
  User Interests Scores Table
Possible Use For Article Recommendation for Historical Articles

ARCHITECTURES
Article Recommendations
Article Recommendation (possible future state)
OVERVIEW

The goal of this playbook is to outline the requirements and steps in order to create an effective Content Recommendation model. By following the steps laid out within this playbook, publishers should be able to maximize the effectiveness of the content in order to impact churn and overall engagement with the content.

There are a few requirements which any publisher should review and confirm feasibility prior to diving into the technical components laid out in the sections following.

BUSINESS REQUIREMENTS

What Would a Publisher Need Prior to Getting Started?

1. A core part of a successful project includes a dedicated team. That team can be made up of any number of people with varying roles, however, we recommend to at least have:
   a. Project manager
   b. An Analyst who deeply understands the Publishers Google Analytics implementation
      i. Note: this could be an agency partner, as long as they are available throughout the project to answer questions and implement components (where required).
   c. Technical lead to assist with any internal development work
      i. Likely the most crucial part of the team. Publishers will want to include someone who is proficient in SQL and who understands the Google Cloud Platform ecosystem intimately.
   d. Internal Data Analyst team
      i. This would be particularly fruitful for post-implementation to create meaningful actionable steps for the publisher to hone in on.

2. Having a clear Goal and Vision of the project output
   a. Depending on your business model, the end goal may vary, however, it should be something that is concrete and outlines how the models should be utilized post-implementation.
      i. For example: “We want to understand the trends, categories, and attributes of our publications readers.”
      ii. Some suggested questions on how to formulate the end goal vision could be:
         ● What are some of the trends that we see seasonally across our readers within our various publications?
         ● Are these categories and content types properly tracked on our sites and reflected in the data?

3. Data
   a. When the initial plan and wishes are established a data audit needs to take place to confirm all key events are being tracked accurately.
      i. It is important that the events are being tracked, however, you’ll also want to confirm the tracking has been in-place and accurate for at least four (4) months.
         ● Why? An accurate Machine Learning needs historical data that is both consistent and in-depth so the more data you have on-hand over an extended period of time, the better.
      ii. The data needs to be important and needs to have consistent naming conventions and indexes for Custom Dimensions and Events across all Properties being analyzed.
4. **Budget**
   
   a. Exact costs for a project like this will vary significantly based on: data size, number of models and their complexity, and supporting infrastructure (external data sources).
   
   i. With this variance, it is hard to suggest a specific budget but Publishers should plan for at least an estimated $250 monthly spend to keep the model up and running.
      
      - Note this cost estimate does not include the NLP API spend required
   
   b. The biggest spender is the NLP API
   
   i. We recommend to evaluate costs related to NLP API. It has a pretty straight forward cost structure and you should be able to estimate monthly costs somewhat accurately.
   
   ii. The costs depend on the number of articles you ran through and their length.
Content Recommendations

Data Collection

Architecture

Data Collection Steps

Refer to the number annotations in the above diagram when reading the processing steps below.

[Step 1] Log Export: Filter

GCP logging allows for exporting copies of some or all of your logs to external GCP components. For example, to publish log message(s) to messaging service like GCP Pub/sub. Exporting involves writing a filter that selects the log entries you want to export, and choosing a destination topic in GCP Pub/Sub. The filter and destination are held in an object called a sink. There are no costs or limitations in GCP Logging for
exporting logs, but the export destinations (pub/sub) charges for storing or transmitting the log data. Additional GCP logging [documentation](#).

**Filter**

```sql
FILTER

resource.type="bigquery_resource" protoPayload.methodName="jobservice.jobcompleted"
protoPayload.serviceData.jobCompletedEvent.eventName="load_job_completed"
protoPayload.authenticationInfo.principalEmail="analytics-processing-dev@system.gserviceaccount.com"
NOT protoPayload.serviceData.jobCompletedEvent.job.jobConfiguration.load.destinationTable.tableId:"ga_sessions_intraday"
```

**Destination**

PubSub topic: ga360-export

**Description**

The log filter above extracts the message that confirm that a `ga_sessions_YYYYMMDD` table has been successfully loaded into BigQuery from Google Analytics. The same consistent email is always used by Google Analytics to export data into BigQuery. The last line of the filter excludes the hourly updated tables with "intraday" as part of the table name. The filter can be expanded to include only certain datasets (GA Views) if so desired but for now all GA views are included and will send log messages.

**[Step 2] Pub/Sub Message: Dataset (GA View)**

Cloud Pub/Sub allows for flexible and reliable enterprise message-oriented middleware providing ingestion and delivery of messages that serve as a foundation for modern stream analytics pipelines. By providing many-to-many, asynchronous messaging that decouples senders and receivers, it allows for secure and highly available communication among independently written applications. Cloud Pub/Sub delivers low-latency, durable messaging that helps developers quickly integrate systems hosted on the Google Cloud Platform and externally.

**[Step 3] Cloud Function: List_article_urls**

Google Cloud Functions are a serverless execution environment for building and connecting cloud services. With Cloud Functions you write simple, single-purpose functions that are attached to events emitted from your cloud infrastructure and services. Your function is triggered when an event being watched is fired. Your code executes in a fully managed environment. There is no need to provision any infrastructure or worry about managing any servers.

This specific function is subscribed to the pubsub topic ga360-export which contains a message with the project, dataset and table that was just loaded into BigQuery (from Google Analytics) or that was simulated from a backfill script. For example: `bigquery-project-name-.100358753.ga_sessions_20190303`. The function has two outputs:

1. Generate a distinct list of GUID’s that have been previously analyzed using Google’s Natural Language API and upload that list to GCP storage for referencing later.
2. A SQL query is then built joining the table in the message and the `bigquery-project-name.nlp_analysis.nlp_articles` table listing which articles URLs needs processing. In order for a URL to be in the results list they must meet the following two conditions:
   1. Be an article (custom dimension equal to 'article')
   2. Not exist already in the NLP results dataset.

Below is the SQL that is executed for a GA View ID example:
The results (list of URL's) from the SQL query are the basis for the loop in which each item (URL) invokes another function called 'analyze_content'.

```
#standardsql
with ga_articles as (  
  SELECT distinct  
      split(hits.page.pagePath, '?')[SAFE_ORDINAL(1)] as url  
  FROM 'bigquery-project-name.<ga-view-id>.ga_sessions_20190303',  
  UNNEST(hits) AS hits  
WHERE  
  hits.type = 'PAGE'  
  and (  
    SELECT  
      MAX(IF(index=2, value, NULL))  
    FROM  
      UNNEST(hits.customDimensions)  
  ) = 'article'  
) select ga.url as url  
from ga_articles ga  
left join 'bigquery-project-name.nlp_analysis_dev.articles' az on ga.url = az.url  
and 100358753 = az.ga_id  
where az.url is null
```

[ Step 4 ] Cloud Function: Analyze_content

The cloud function called analyze_content also has 2 inputs:
   1. A list of distinct GUID’s that already had NLP processing and results stored in BigQuery.
   2. The GUID from the CMS object by appending "?template-jsonart&mime-json&omniture-0" to the URL and executing a GET request.

Armed with the URL and GUID, the GUID is compared to the list of (#1 above) distinct GUID’s. If a match exists, only an entry into the articles tables will proceed. If it does not exist, NLP analysis will be applied to the title, summary and body of the article and both an insert into articles and nlp_results BigQuery tables will occur.

The powerful pre-trained models of the Natural Language API let developers work with natural language understanding features including sentiment analysis, entity analysis, entity sentiment analysis, content classification, and syntax analysis. Natural Language uses machine learning to reveal the structure and meaning of text. You can extract information about people, places, and events, and better understand social media sentiment and customer conversations. Natural Language enabled us to analyze text from the articles published and classify them. We ran 10 days of viewed articles from the 16 in-scope websites and found the average costs for different NLP analysis. In the end, we elected to only run the content classification analysis because the downstream processes were only using the results from it to make content recommendations. It is possible at a future date to add additional analysis and store the results in BigQuery.

<table>
<thead>
<tr>
<th>NLP Analysis Type</th>
<th>Cost per article</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity Sentiment</td>
<td>$0.0097</td>
</tr>
<tr>
<td>Entity Analysis</td>
<td>$0.0048</td>
</tr>
<tr>
<td>Sentiment Analysis</td>
<td>$0.0048</td>
</tr>
<tr>
<td>Content Classification</td>
<td>$0.0038</td>
</tr>
<tr>
<td>All possible Analysis</td>
<td>$0.0231</td>
</tr>
</tbody>
</table>

The above are average costs of NLP analysis based on 10 days of articles. Actual NLP pricing from google.

[Step 6] Google BigQuery: nlp_analysis

BigQuery, Google’s serverless, highly scalable enterprise data warehouse, is designed to make data analysts more productive with unmatched price-performance. Because there is no infrastructure to manage, you can focus on uncovering meaningful insights using familiar SQL without the need for a database administrator.
Big Query Datasets

The screenshot below shows the NLP datasets.

The articles table holds a record of which URL’s have been processed and their status. The GUID column is used to join to the nlp_results table to retrieve the NLP results associated with that URL/GUID. Below are the table schemas.

**BigQuery Table: Articles**

<table>
<thead>
<tr>
<th>Field name</th>
<th>Type</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>STRING</td>
<td>NULLABLE</td>
<td></td>
</tr>
<tr>
<td>time_stamp</td>
<td>TIMESTAMP</td>
<td>NULLABLE</td>
<td></td>
</tr>
<tr>
<td>guid</td>
<td>STRING</td>
<td>NULLABLE</td>
<td></td>
</tr>
<tr>
<td>geo_name</td>
<td>STRING</td>
<td>NULLABLE</td>
<td></td>
</tr>
<tr>
<td>geo_id</td>
<td>INTEGER</td>
<td>NULLABLE</td>
<td></td>
</tr>
<tr>
<td>status</td>
<td>STRING</td>
<td>NULLABLE</td>
<td></td>
</tr>
<tr>
<td>keywords</td>
<td>RECORD</td>
<td>REPEATED</td>
<td></td>
</tr>
<tr>
<td>keywords.keyword</td>
<td>STRING</td>
<td>NULLABLE</td>
<td></td>
</tr>
</tbody>
</table>
BigQuery Table: nlp_results

<table>
<thead>
<tr>
<th>Field name</th>
<th>Type</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>guid</td>
<td>STRING</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>time_stamp</td>
<td>TIMESTAMP</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>sentiment</td>
<td>RECORD</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>sentiment.score</td>
<td>FLOAT</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>sentiment.magnitude</td>
<td>FLOAT</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>categories</td>
<td>RECORD</td>
<td>REPEATED</td>
</tr>
<tr>
<td>categories.name</td>
<td>STRING</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>categories.confidence</td>
<td>FLOAT</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities</td>
<td>RECORD</td>
<td>REPEATED</td>
</tr>
<tr>
<td>entities.name</td>
<td>STRING</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.type</td>
<td>STRING</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.saliency</td>
<td>FLOAT</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.sentiment</td>
<td>RECORD</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.sentiment.score</td>
<td>FLOAT</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.sentiment.magnitude</td>
<td>FLOAT</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.mentions</td>
<td>RECORD</td>
<td>REPEATED</td>
</tr>
<tr>
<td>entities.mentions.sentiment</td>
<td>RECORD</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.mentions.sentiment.score</td>
<td>FLOAT</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.mentions.sentiment.magnitude</td>
<td>FLOAT</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.mentions.type</td>
<td>STRING</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.mentions.text</td>
<td>RECORD</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.mentions.text.beginOffset</td>
<td>INTEGER</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.mentions.text.content</td>
<td>STRING</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.metadata</td>
<td>RECORD</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.metadata.wikipedia_url</td>
<td>STRING</td>
<td>NULLABLE</td>
</tr>
<tr>
<td>entities.metadata.mid</td>
<td>STRING</td>
<td>NULLABLE</td>
</tr>
</tbody>
</table>
User Interests Scores

Based on the data we have from the NLP analysis we can build additional tables containing data around user interests and recommendations.

Creating User Interest Scores

Every article that we process through NLP API gets classified which means it gets labeled with one or more of the possible categories shown [here](#).

This gives us an idea of what the content of the text represents, so with this information we can group articles in category groups based on their content itself.

So first ingredient in figuring out user interests are the content categories which we have for each article. But we also have all articles that were read by each user either anonymous or a known user. For each type of user we can create a historical path of all articles read up to the current point.

On one side we have the categories and article URLs and on the other side we have the users and articles that they've read. So the common key between the two is the article URL.

For calculating interest scores we used a well known approach called [TF-IDF](#). This is an approach commonly used to figure out what words are important in certain text documents. In our case we repurposed the word document, which in our text means all the article categories a user encountered during their lifetime.

You can think of a user as a document of all the different categories they visited with added time component meaning if a user read an article long ago, those categories will count a little less and the newer ones count more.

This helps us keep the calculation as recent as possible relying more on the article categories recently visited.

On a high level, TF-IDF does two things:

1. First, it determines how important the category is for them specifically. It looks at all categories and the one they’re most interested in is the one that pops up the most.
   a. The problem with that is, maybe the site is about basketball and most of the articles are about basketball, so in the majority of users basketball would be the top category.
2. This is the reason why there’s a second part, which looks at how often does that category show up in other users (documents). If a category is very common then it will have a lesser weight because it doesn’t stand out, but if the category is rare then it will have a higher weight.

When combining the two sides we get a balanced calculation for each user and their interest in a specific category.

The final table includes interest scores for every known user (CD X ID can be found), and it also includes interest scores for every anonymous user with at least 2 sessions. We look at data from 2019 forward.
## User Interests Scores Table

### Table Details: user_category_interests

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Nullable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timestamp</td>
<td>TIMESTAMP</td>
<td>NULLABLE</td>
<td>Timestamp of the interest calculation</td>
</tr>
<tr>
<td>propertyName</td>
<td>STRING</td>
<td>NULLABLE</td>
<td>Publication name</td>
</tr>
<tr>
<td>identifier</td>
<td>STRING</td>
<td>NULLABLE</td>
<td>GA = anonymous user, PLATFORM = known user (Piano/MG2)</td>
</tr>
<tr>
<td>userId</td>
<td>STRING</td>
<td>NULLABLE</td>
<td>GA client id or Piano/MG2 id</td>
</tr>
<tr>
<td>category</td>
<td>STRING</td>
<td>NULLABLE</td>
<td>NLP category</td>
</tr>
<tr>
<td>lastVisitStartTime</td>
<td>TIMESTAMP</td>
<td>NULLABLE</td>
<td>Last visit of a user for that category</td>
</tr>
<tr>
<td>sessions</td>
<td>INTEGER</td>
<td>NULLABLE</td>
<td>Number of sessions the category was seen by the user</td>
</tr>
<tr>
<td>articles</td>
<td>INTEGER</td>
<td>NULLABLE</td>
<td>Number of articles the category was seen by the user</td>
</tr>
<tr>
<td>totalSessionsWithArticles</td>
<td>INTEGER</td>
<td>NULLABLE</td>
<td>Number of total sessions with article views for a user</td>
</tr>
<tr>
<td>totalArticles</td>
<td>INTEGER</td>
<td>NULLABLE</td>
<td>Number of total articles read seen by a user</td>
</tr>
<tr>
<td>totalUniqueArticles</td>
<td>INTEGER</td>
<td>NULLABLE</td>
<td>Number of total unique articles seen by a user</td>
</tr>
<tr>
<td>interestScore</td>
<td>FLOAT</td>
<td>NULLABLE</td>
<td>Interest score for the category for the user</td>
</tr>
</tbody>
</table>
Possible Use For Article Recommendation for Historical Articles

1. **User Article Path** (BigQuery)
2. **NLP Categories + Article URL** (BigQuery)
3. **User + NLP Categories** (BigQuery)
4. **User Category Interests** (BigQuery)
5. **Similar Users** (BigQuery)
6. **Article Viewed Last Week** (BigQuery)
7. **Top N Recommended Articles For a User** (BigQuery)

Process:
- User Article Path and NLP Categories + Article URL are merged using URL as a join key.
- User + NLP Categories are processed using TF-IDF.
- User Category Interests are derived from the processed data.
- Similar Users are identified based on user category interests.
- Article Viewed Last Week is considered for recommendation.
- Top N Recommended Articles For a User are generated based on the above processed data.
**Note**

NLP processing happens every time a new `ga_sessions` table is loaded into BQ. This means we are always one day late at processing articles and can therefore recommend only articles from the previous day.

RSS Feed checking solves that, because as soon as the article is published we would NLP process it and store to BQ.
Article Recommendation (possible future state)

Have Questions? Reach out!
Adswerve is a team of 200+ industry leaders using data discovery to help our clients create insightful marketing strategies. As a company, Adswerve is obsessed with bringing life to data and illuminating human connections for hundreds of agencies, analysts and marketers—solving complex challenges for some of the biggest brands in the world. Adswerve is a Google Cloud Marketing Analytics, Google Analytics, and Google Cloud Premier Partner and can help business both large and small execute on projects such as this. Reach out to our team of experts to learn more.
About Adswerve

Adswerve is a team of industry leaders who believe smart marketing is built on data discovery. Some of us are former Analytics Pros data scientists, others are media experts. We’re all obsessed with bringing life to data and illuminating connections for hundreds of agencies and some of the world’s biggest brands. Learn more at www.adswerve.com.

About The Google News Initiative

The Google News Initiative is our effort to help journalism thrive in the digital age through evolving business models to drive sustainable growth, elevating quality journalism and empowering news organizations with new technology.

For more information visit:

www.adswerve.com

g.co/newsinitiative