Product or Service Research Assistant

Google LLC
Product or Service Research Assistant

This document is directed to a product or service research assistant that helps you choose which product to buy from the many options available. Although this document will focus on a product research assistant, each idea also applies to a research assistant user interface (UI) for services professionals, travel destinations, restaurants or other structured entities and should be considered in the scope.

The user begins by selecting a particular product type that they want to research (for example, stroller, TV, vacuum cleaner, etc.). This may be done by choosing between a range of displayed options, through a search box, through a category browser or some other method. Once the user has chosen a product category, they are shown a) all the products within that category available for purchase (possibly limited by only those available in a particular country, or other market limitations) b) a range of dimensions by which those products can differ (e.g. for strollers, whether they have 3 or 4 wheels, whether they're double or single, the height of the handle, etc.) - these are noted as facets in the rest of the document c) a way to sort, refine and modify the list of available products to allow the user to choose which product is right for them. The particular UI features and algorithmic implementations that allow the user to find the correct product are listed below.

Additionally, the user may click on a product to see a product page with more information about that specific product. It may include photos, videos, text, technical data, color or other configuration options, user reviews, expert reviews, common review themes, user star (or other quality) ratings, expert star (or other quality) ratings, certifications, hybrid ratings, awards won, product manuals, safety information including recalls, prices, places to buy, additional price information like whether tax or shipping will be charged, physical locations where the product may be purchased, physical locations where the product may be viewed on the showroom floor for in-person testing, a map and the ability to show the nearest physical locations to the user's location (or another location of interest), coupons available, or any other information that a user may use to make a decision about whether or not the product fits their needs. This data could be created by a retailer, shopping service provider, crawled from across the internet, discovered manually, or syndicated from others.

The tool maintains state at all times, persisting the user's shortlist and research choices. The technology and user interface for this is described more below. The user can hide/favorite/promote/annotate/configure or otherwise interact with each product option available. The user may invite others to collaborate with him or her, and that collaborator may be able to edit the existing research project. The user may also invite other users to view his or her research, and that second user may elect to begin a new research project from the starting point of the first users research.

The tool may display brand pages that list information about brands, similar to the product pages. Those pages may display news about the brand or people relevant to the brand (e.g., the CEO), products the brand sells, aggregate information about that brand's products (e.g., number or quality of safety recalls across their product suite), brand values as perceived by users, brand values as professed by the brand itself, brand imagery, brand activities like charity or community service reach outs, brand origin/founding stories, certifications or awards at the
brand level, videos about that brand. This data could be created by retailers, shopping service providers, crawled from across the internet, discovered manually, or syndicated from others. This data could also be provided by the brands, and shopping service providers could elect to charge brands for placement on the brand pages (or elsewhere on the site).

Shopping service providers may allow third parties to promote coupons or other deals to users who are at particular stages of their research process or interested in particular product types, brands or models. Shopping service providers may allow specific retailers to promote coupons or other deals to users who have expressed specific interest in buying a product at that location in the product page (or some other method).

The tool may exist on the web, a desktop application, a mobile application, a tablet application, a smartwatch application, a home hub application or an application on any other platform.

3-valued continuous facet

When filtering a set of search results by a continuous variable (e.g. price), shopping service providers allow the user to set an ideal value in addition to a maximum and minimum value. Setting a maximum value filters out any results that are above that value. Setting a minimum value filters out any results that are below that value. Setting an ideal value does not filter out results but instead re-ranks the displayed results by preferring results that have values close to the ideal value (although this signal is weighted along with other ranking factors, including other ideal values that have been set in other facets).

The maximum and minimum may be automatically calculated based on the existing result set. The maximum and minimum "handle" or endpoints may be restricted to only be set to a fixed range of values based on some minimum increment across the continuous variable (for example, the minimum increment for a price facet might be $10). The minimum increment may change across the range of the facet (for example, the last increment may be bigger to handle significant outliers). A bar graph or similar indicator may be displayed to show the user how many results they will be filtering out or adding when they move the maximum or minimum handles or endpoints.

User interface and interactions

Default view:

Handles are set to either edge
- The text shows lowest and highest values for the result set. Input field for ideal is visible and any value can be typed.
- The height of the bars in the graph shows number of available products per increment.

On Click:
- Clicking anywhere in the slider area displays the incremental values.
- Sliders can be dragged in increments.

Setting the min. and max. values:
- As the sliders are being dragged, the actual corresponding value is displayed
- The lowest and highest values will remain on either edge, in a disabled (greyed out) state.
- The active part of the graph gets updated as the handles are moved; ranges outside of min. and max. are greyed out.

Setting the ideal:
- An ideal value can be set by typing into the text input box.
- The arrow indicates the position of the ideal value on the slider bar.
- Once an ideal value has been set, it also can be changed by dragging the arrow. This will update the number in the text input field.
- If the ideal is outside the previously set min. and max. values, those will expand and be updated (pushed out) accordingly.

Technical implementation

1) Indexing and calculation of bar graph dimensions

```
Index documents into search index with facet as an field in the document

Compute and store min, max values for facet in database (DB) or key-value store (KVS)

Compute histogram and store min, max values in DB or KVS
```
2) Run-time flow

Boost checkbox facets

When filtering a set of search results by a binary variable (e.g. is on sale), shopping service providers allow the user to set the facet as a Nice To Have or Don't Want in addition to Must Have or Don't Care.

If the user sets the facet to Must Have, the search results are filtered to only those products with the value. If the user sets the facet to Don't Want, the search results are filtered to only those results without the value. If the user sets the facet to Nice To Have, the results are not filtered but instead re-ranks the displayed results by preferring results that have that value (although this signal is weighted along with other ranking factors, including other Nice To Have values...
that have been set in other facets). If you user sets the facet to Don't Care, the presence or absence of the value is not taken into account when ranking the results.

The checkbox may work as a standard checkbox, allowing users to toggle between Must Have and Don't Care by clicking in the checkbox. Alternatively, a user may bring up a menu of options, including Nice To Have and Don't Want, and select their preferred value from there. The menu could appear automatically on hover over the checkbox, or be displayed by the user clicking a drop down or similar.

User interface and interactions

Unchecked (default):
- The default state of all checkboxes is unchecked.

On Hover:
- If the user hovers for some length of time (e.g. 2 seconds) over a checkbox, a pop-up will be displayed with the three options. Icons and labels are clickable.

On Click:
- If the user clicks on a checkbox directly, the checkbox changes to the "must have" state.
- The selected state of the checkboxes are based on the choices a user has made.
- Clicking on a selected checkbox brings back the default (Don't Care) state.
- Hovering over a selected checkbox for some length of time will display the on hover state.
- Instead of a hover, the checkbox options may be displayed by selecting from a drop down menu.
### Technical implementation

![Diagram of the technical implementation process](image)

- **Client sends current boosted checkbox setting to search service.** Checkbox can be in one of four states.
- **Search service maps settings to search query in the following way:**
  - Don't Want & Must Have: boolean filters
  - Don't Care: no-op
  - Nice To Have: search result scoring boost
- **Search service sends client results**

### Slider multi-valued facets

When filtering a set of search results by a multivalued variable (e.g. color), shopping service providers allow the user to set each value to Don't Include, Include, and Boost using a series of sliders.

If the user sets a value to Don't Include, all products matching that value are filtered out. If the user sets a value to Include, all products matching that value are included in the result set. If the user sets a value to Boost, all products matching that value are included and the results are re-ranked to prefer the results that match that value (although this signal is weighted along with other ranking factors, including other Boost values that have been set in this or other facets).

Users may be able to set all sliders for a single facet to Include in one set by clicking a reset button (e.g. show all colors). Users may be able to set all sliders for all but one value (e.g. show me only pink) to Don't Include by setting the slider to Boost, then clicking a Set Only button that may display for a short time.

### User interface and interactions

**Default:**
- All sliders are set to center (Include)
- Moving a slider to the left switches off that attribute (Don't Include).
- Moving a slider to the right boosts results with that attribute (Boost)
- The slider bar may change color to indicate the selection (e.g. brighter for Boost and more subdued for Don't Include)

**Boost:**
- Moving a slider to the right also shows "only" as an additional button. Hovering over the icon may show a tooltip (e.g. "Show Only This").
- If the mouse moves away from the button, it fades out.

**Clicking on "only":**
- If the user has clicked "only" for one options, the sliders for all other options move to the left (Don't Include) and the icon disappears.

**Moving another slider to "all":**
- If another slider gets moved to the right, the "only" icon for that slider will be displayed.
- The icon disappears when the mouse has been moved away.
- Clicking on a slider that is already set to "all" will bring up the "only" icon again.

**Technical implementation**

Client sends current sliders setting to search service. Each slider can be in 1 of 3 states ("Only" is a client-side treatment whereby all other sliders are set to the "Don't include" state).

Search service maps settings to search query in the following way: Don't include: boolean filter Include:no-op Boost: search result scoring boost

Search service sends client results

**Promoted facets**

Shopping service providers may allow third parties to pay to promote certain facets in the facet bar. For example, if a brand releases a new particularly lightweight car seat, they may choose to promote the weight facet to the top of the car seat facet list. Prices may be set via an auction.
**Wizard/questions in combination with search results and facets**

Questions or forms - either individually or as a wizard flow - may appear at the top of the search results. The user's input is translated to facet settings (and thus ranking of search results). The questions or wizard allow the user to navigate the range of technical choices in a simplified and less overwhelming manner.

**User interface and interactions**

On load, questions may appear at the top of the results:

![Modal window](image1.png)

When clicked, a modal may appear to provide the user with information and solicit their input:

![Modal window](image2.png)
After interacting with the modal, the facets update to reflect the user's wishes, and the results also update accordingly. There may be some indication that the user has provided input for an item.

Preliminary context combined with facets

Questions or brief text or explanatory images may appear over a facet to provide context for the technical feature. The context is primarily to help the user understand if the facet matters to them or not. The context may also serve to simplify the facet display and make it less overwhelming to users.
User interface and interactions

On load:

- Facets are displayed as a list of context questions that place the technical value in the facet in the context of the user's life
When the user clicks the > it unveils the facet setting:
**Hiding and favoriting search results**

Rather than a static list of results, allow users to remove results that they are not interested in from the result set. This allows the user to narrow down their selections to the ones that they're actually interested in. These results stay hidden even after subsequent adjustments of the facets.

To expedite the narrowing down of search results, users can also select a few results as their favorite, then toggle between displaying all results (that haven't been hidden) and displaying only their favorites.

**User interface and interactions**

Single result with options to hide or favorite:

![Product Name
by Brandname
$500
BUY](image)
Single result that has been favorited:

![Product Image](image_url)

**Product Name**
by Brandname

$500

BUY

Results showing both favorites and non-favorites:

[Shortlist Image]
Results with "show favorites only" toggle on:
Technical implementation

1) Upon hiding:

Client sends the 10 of the item to be hidden to search service

Search service stores the hidden item with the research project 10, a hidden state, and timestamp in a DB or KVS

Client re-issues search query

Search service retrieves all IDs marked with a hidden state for this project and filter them from the search result

2) Upon unhiding:

Client sends the 10 of the item to be unhidden to search service

Search service updates the record of the hidden item with an unhidden state and timestamp in a DB or KVS
3) Upon favoriting:

Client sends the ID of the item to be favorited to search service

Search service stores the favorite item with the research project ID, a favorited state and timestamp in a DB or KVS

Client re-issues search query

Search service retrieves all IDs marked with a favorited state for this project and include them via an OR filter in the search query

4) Upon unfavoriting:

Client sends the ID of the item to be unfavorited to search service

Search service updates the record of the favorited item with an unfavorited state and timestamp in a DB or KVS

Collaboration and history

Collaboration features allow users to make purchases that require the input of two or more people (for example, between partners or roommates for the home or to solicit the opinion of a stylist or interior designer). Features include the ability to add another user to a research project, to maintain research project state across many users, and to keep track of users' past actions so that a different user can always trace back and see how the research has been changed by their collaborator(s).
User interface and interactions

The initial user may invite a new user to collaborate

Invites may be received by email (or other means) and include a link that will automatically add the user to the research project. The link may include protections to ensure that links cannot be spread inappropriately.

Edits to the research shortlist may appear as history items attributed to the user who performed them.
The history may also show how long the research project has been active for, as well as displaying the currently collaborating users, and allow the users to set a title and/or notes for the project:

Technical implementation

1) Single-user scenario

Client gets initial server-assigned user ID and research project ID after initial handshake

Server stores the following in DB or KVS on an ongoing basis:
1) complete snapshots of research project state against the project ID (e.g. user settings on facets, search query, hidden and favorite results)
2) history objects (user actions - e.g. search refinement or hiding - and affected results) against the user ID and project ID
3) hidden/unhidden and favorite items against the user ID and project ID
2) Upon session sharing

Client solicits from the user the email and name of the recipient. Client also solicits from the user her own name and email if they have not previously been given.

Client sends first user's ID, first user's name, first user's email, recipient's name, recipient's email to the server

If recipient email already exists in DB or KVS, corresponding user ID is fetched. If recipient email does not currently exist in DB or KVS, new user ID is generated. Second user ID is associated with the current research project ID.

Server sends second user ID to client

Client generate link with second user ID and research project ID. Client sends invitation to second user with generated link.
3) Multi-user scenario

Dynamic snippets and compare view

Online a-commerce search results pages often allow users to select a small number of results and generate a compare view on a separate page that allows the users to compare those products along a number of dimensions. However shopping service providers present a novel UI that turns the search result page, or a similar research project view, into a dynamic compare view without overwhelming the user with too much information. A snippet of text describing each aspect or facet of the product may be dynamically switched on or off based on their relevance to the user and their purchasing decision. A snippet may be automatically turned on if the user expresses that that facet or aspect is relevant to their purchase decision in another way (by e.g. interacting with a filter facet) or the user may toggle the display directly.

Additionally, the snippets may be dynamically highlighted based on how closely that aspect reflects the user's "Nice to Have", "Ideal" or "Boost" settings in the search refinements (aka facet bar). This may be a direct highlighting of text that matches the user-specified criteria, or it may be a score indicating the number of matches, or similar.
User interface and interactions

Showing the base case with all dynamic snippets turned off:
Results with two dynamic snippets toggled on (weight and price) and highlighting of both prices and weights that are near to the ideal values. The eye icons next to the facets toggle the snippets on and off.
Instead of displaying highlights, the number of matches to the ideal/nice-to-have/boost criteria may be shown:
Technical implementation

Automatic extraction of schematic product information

Shopping service providers extract structured product information in order to enable searching, filtering, and ranking based on facets. Product schemas defining product features are created a priori (editorially or crowd-sourced or machine-learned) and facets are combinations of one or more product feature.
Examples of product features include but are not limited to dimensions, color, material, camera resolution, and safety rating. These product features can be used directly as facets as appropriate for the product domain. Other examples of product facet include but are not limited to "targeted audience" for the product (e.g. for hobbyists vs professionals), which can be a combination of features such as camera resolution and weight and battery life, and "environmental friendliness", which can be a combination of features such as material and production method.

Since the majority of product descriptions and specifications come only in semi-structured formats and provide only the most basic of product features in schematized form, shopping service providers created a feature extraction pipeline to derive the product features and subsequent facet values. Once the required features and taggers are designed (with human input), these features are automatically extracted as new products are discovered and processed. Where automatic tagging falls short, editorial clean up can be incorporated into the pipeline.
Technical implementation

Product Information is sourced from a variety of sources (e.g. data feeds, crawling, user-submitted data or correction) and format (e.g. html pages, social media content, pdf files, videos).

Content-specific data processing pipeline is run on each source, e.g. for text, part-of-speech tagging, stemming, synonyminyng

Feature-specific taggers are constructed for each source, e.g. rule-based recognizers, classifiers, ontology look-up.

Each tagger assigns one or more feature-value-confidence triplets to the document.

Feature-values are conflict-resolved and consolidated based on confidence level from each tag. Resolution strategies are domain specific.

Feature-value expansions based on ontology are needed in some cases (e.g. steel and aluminum are kinds of metal, salmon and peach colors are aliased with orange)

Cleaned up feature-values are used to create facet values for search based on pre-defined mapping

Online search result diversification based on product attributes

Traditional information retrieval applications (e.g. internet searches and product searches) typically order results via some combination of relevance and authority scores, typically in
descending order. Relevance refers to the closeness in information content between the result and the search query and authority generally refers to the varying notions of trustworthiness of the result. Consequently, the "best" search results typically appear close to the top and search results degrade in quality as more results are returned.

Other information retrieval applications such as product search, however, could benefit from not ordering the results by the above-mentioned scores but rather to enable product discovery by showing a large variety of results close to the top. Shopping service providers created a system which satisfies these two basic principles: (1) When a user initially arrives at the product browse pages or performs a categorical search, the system displays the full landscape of available products. (2) As the user refines her searches within the session, subsequent results are diversified within the constraints of the search query.

To illustrate with an example of a car buying site, an initial list of products may include sedans, SUVs, convertibles, minivans of varying makes, engine sizes, and colors. If the user refines their search to include only black SUVs (either via textual queries or structured preference entry), the results are then limited to SUVs in black but are still diversified across makes and engine sizes.

Technical implementation

1) Offline diversity feature definition

| Editorially define an ordered list of features important for diversification, e.g. for cars, car types (sedan, SUV) > make > engine size > color |

| Product features are extracted for each product (e.g. via the procedures described in the “Automatic extraction of schematic product information” section) |

| Index searchable documents (e.g. products) into search index along with all the feature dimensions which can be diversified |
2) Online search result diversification

The greedy algorithm employed largely determines the nature of the final results. One example algorithm is to perform a round-robin selection from each partition. Alternatively, a scoring framework can be constructed so that the algorithm can greedily maximize the total diversity score. For example, each additional distinct value in a diversity feature contributes a decreasing amount to the score (the intuition is that including a 3rd car type in the results is significantly useful to the user, but including a 20th car type is only marginally helpful). The algorithm then selects the next best result to include in the set based on some combination of the diversity score and the original search result score.

Influencing search result diversity via offline search result ordering

While diversity in results is valuable in a number of information retrieval applications (e.g. for recommendations or for presenting a wide variety of choices when user intent is unclear), result diversification is typically performed as a re-ranking step online and can lead to additional latency in response time.
Shopping service providers created a method for which an offline diversification algorithm can first be used to determine an a priori ordering of the search results. In the case where user intent is vague and performs a categorical search, the final ordering of the search results is largely driven by the a priori ordering. However, it exerts decreasing influence on the final ordering of results as the search queries become more refined within a session. This can be accomplished via a tuned search ranking/scoring system where the scores become proportionally dominated by each satisfied search parameter.

Technical implementation

1) Offline diversity-driven result ordering

- Editorially define an ordered list of features important for diversification. e.g. for cars, car types (sedan,SUV) > make > engine size > color

- Define a diversity scoring framework based on the diversity features. *Same discussion as in the online diversification algorithm applies.

- Initialize the ordering either with a random result or an editorially featured result.

- Generate a linear ordering of the remaining search documents by using a greedy algorithm to select the next document that maximizes diversity score (see discussion).

- Index searchable documents (e.g. products) into search index as usual but include the diversity rank order as a new field.
2) Online search result diversification

User issues a search query for a product (e.g. via a client)

Search service performs search query on the search index to generate a recall set

Search index scores the documents such that the diversity rank order carries a certain weight, but each additional matched search term I facet carries additional weight (see discussion).

The relative weight between the diversity rank order and the matched search terms I facets determines how quickly the search results become focused on query relevance as the user search intent becomes clearer. The tuning is application-specific.

**Generation of result facet counts based on modifiable facet schema**

In order to provide cues to the user as to how the number of search results may be affected by their facet preferences, shopping service providers created different visual indications for number of current search results that satisfy each facet value.

The facets are modifiable over time and the total number of searchable results may also fluctuate. Shopping service providers therefore created a system that accepts the facet specifications as a parameter and dynamically map the facet structure to search queries and utilizes it to generate search result counts.
Technical implementation

Client issues search query to search service containing user's faceted preferences

Search service maps the preferences into a search query based on the facet schema and performs it on the search index to generate a recall set.

Stats are computed as follows for each facet defined in the schema: For numerical facets (e.g. 3-slider preferences), compute a histogram over buckets pre-defined in facet schemas and the bounds (max height, min value, max value). For binary facets (e.g. boosted checkboxes), compute counts of true and false values. For multi-valued facets (e.g. multi-valued sliders), compute counts of each value in the range.

Search service sends counts along with search results to client.

Progressive user sign up and search session management

Shopping service providers allow our users to begin using our research tool anonymously while maintaining their research project states even as they leave the site or app and return at a later point in time. When users come back to the site or app, their facet preferences, hidden items, favorited items, project titles and notes are all restored. Users are subsequently invited to provide additional personal information as needed, e.g. if they want to invite someone else to collaborate on the project or to work on the same research session across devices.

Shopping service providers manage this by generating IDs for each anonymous user and each project and storing research project data against these IDs. The generated IDs do not contain personal information and are stored as session cookies or app local data, allowing the user to access their last research project state.
Technical implementation

1) Upon new user visiting the site

- Client sends handshake request to server with no user information

- Server assigns a new randomly-generated user ID and a new randomly-generated research project ID.

- Server stores the newly generated research project ID in DB or KVS along with the complete search research project state (e.g. query parameters containing user preferences, hidden items, favorited items, notes). The newly generated user ID is associated with this project ID.

- Client stores User ID and research project ID locally (in the browser as a cookie or in the app)
2) Upon user returning to the site

Client looks up locally stored User ID and research project ID. If found, they are sent along with handshake request to server.

Server retrieves last research project state associated with the project ID and sends to client.

Client renders last project state to user. User continues refining research project.

Server stores each subsequent snapshot of project state against the user ID and research project ID in DB or KVS.

3) Upon user volunteering additional personal information

Client sends additional user information (e.g. name, email) to the server

Server updates user information in DB or KVS

Review and/or review snippet ranking based on "contentfulness"

User product review content is often vast and contradictory, making it difficult to parse quickly. This problem may be simplified by extracting key themes and/or ranking the reviews - or the snippets themselves - by how representative they are of user sentiment. This may include taking into account how many reviewers express a similar sentiment about a particular product aspect, and the degree of specificity or uniqueness that sentiment about a particular product aspect has.
Technical implementation

Automatic extraction of schematic subjective product ratings

User product review content is often vast and contradictory, making it difficult to parse quickly. This may be solved by parsing all review content (including the text and star review score) and extracting out one or more scores. This may be as simple as an overall goodness or quality score, or may be as complex as set of scores on a number of dimensions that are particularly important to that product type. For example, if the product category was rear facing baby car seats, the dimensions might include safety, ease of installation, and lightweight.
Technical implementation

1) Creation of "gold standard" corpus

Editorially define a list of subjective rating dimensions for the chosen category of products (e.g. durability, quality, aesthetics, comfort)

Source user reviews (via feeds, crawlers, etc) for a selection of products, including the review title and text, star ratings, and helpfulness score, if available.

For each review, create training data by annotating whether the reviewer expressed positive, negative, or no sentiment regarding each subjective rating dimension.

Optionally also delimit the portion of the review that is relevant for each subjective rating dimension for the non-neutral sentiments.

2) Classifier training via machine learning

Convert gold standard corpus into training data by extracting features from each annotated review. Features include but are not limited to:
- N-gram counts in a bag-of-word model
- N-gram counts normalized by Inverse document frequency (IDF)
- Word counts for words in sentiment dictionary
- N-gram counts with synonym expansion
- Review star rating

Learn one classifier per subjective rating dimension on the 3-way classification task (positive, negative, no sentiment)

Since a wide variety of machine learning techniques apply here (e.g. logistic regression, SVM, decision trees, random forest, ensemble methods), select the best performing classifiers based on cross validation and/or ROC curve.
3) Tagging of new products

Source user reviews (via feeds, crawlers, etc) for new products or products for which no gold standard data exist, including the review title and text, star ratings, and helpfulness score, if available.

Extract the same features per review as used for training the classifiers.

For each subjective rating dimension, run the corresponding classifier on each product review to detect positive or negative or no sentiments.

The overall sentiment in each dimension is the ratio of positive to negative sentiments detected.