Query Understanding for Surfacing Under-served Music Content

Federico Tomasi, Rishabh Mehrotra, Aasish Pappu, Judith Bütepage, Brian Brost, Hugo Galvão and Mounia Lalmas
Spotify
{federicot,rishabhm,aasishp,judithb,brianbrost,hugog,mounial}@spotify.com

ABSTRACT

Platform ecosystems have witnessed an explosive growth by facilitating interactions between consumers and suppliers. Search systems powering such platforms play an important role in surfacing content in front of users. To maintain a healthy, sustainable platform, systems designers often need to explicitly consider exposing under-served content to users, content which might otherwise remain undiscovered. In this work, we consider the question when we might surface under-served content in search results, and investigate ways to provide exposure to certain content groups. We propose a framework to develop query understanding techniques to identify potential *non-focused* search queries on a music streaming platform, where users' information needs are non-specific enough to expose under-served content without severely impacting user satisfaction. We present insights from a search ranker deployed at scale and present results from live A/B test targeting a random sample of 72 million users and 593 million sessions, to compare performance of different methods considered to identify *non-focused* queries for surfacing under-served content.

CCS CONCEPTS

• **Information systems** → **Query intent**; *Query representation.*

KEYWORDS

query understanding; music recommendation; under-served content

ACM Reference Format:
Federico Tomasi, Rishabh Mehrotra, Aasish Pappu, Judith Bütepage, Brian Brost, Hugo Galvão and Mounia Lalmas. 2020. Query Understanding for Surfacing Under-served Music Content. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3340531.3412741

1 INTRODUCTION

Modern platforms, such as Uber, Amazon, Airbnb, Spotify and YouTube, are increasingly emerging as the go-to applications facilitating economic exchange between consumers and suppliers. These platforms need to meet the needs not only of the demand side (*e.g.*, users), but also on the supply side (*e.g.*, retailer, artists). In the success of these platforms, search functionality plays a key role, as it links the direct needs of the users to relevant content provided by the suppliers.

Most traditional search systems have been user-centric in their approach of optimization and evaluation. Systems that power multi-stakeholder marketplaces however have to account for supplier exposure on their platforms. Typical of most search and recommender systems, there often exists content that is not surfaced to the users. This can be caused by a lack of metadata, popularity bias or unspecific query formulation. Further, blindly optimizing for consumer relevance has shown to have a detrimental impact on supplier exposure and fairness [22]. Indeed, an ill-optimized search system might unintentionally provide differential exposure to certain set of suppliers. To ensure a healthy, sustainable platform, systems designers need to consider this disparate impact on supplier exposure and develop ways of mitigating it.

Given the important role search plays in surfacing content, investigating ways to provide exposure to certain content and supplier groups via search results would go a long way in giving system designers more control over consumption on their platform.

In this work we consider the question: when might we surface under-served content in search results? Specifically, we consider a search system powering a large scale music streaming platform, in which artists pose as suppliers and users pose as customers. Recent findings around the non-specificity in user intents [19, 21] indicate that many opportunities to present under-served content exist on music streaming platforms. Users often have low or no preference about the specific content they wish to stream. Such opportunities can be exploited to surface under-served content, thereby enabling that content to reach a broader audience.

We propose a framework to develop query understanding techniques to identify potential *non-focused* search queries on a music streaming platform [19]. We posit that there exists certain *non-focused* queries with broad intents, where users are generally more open to non-specific recommendations (*e.g.*, "relaxing music"). Identifying such *non-focused* queries provides opportunities to surface result from undiscovered, under-served music content. This can help platforms to improve exposure of under-served content, without impacting user satisfaction.

To instantiate the problem and identify *non-focused* queries, we assume two types of content we wish to surface more, which we use as running examples throughout this work.

Niche Genres: Typical of most recommender systems, there exist categories of content not yet discovered by a broader audience. For example, genres like Classical, Jazz and Blues have historically been

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-6859-9/20/10…$15.00
https://doi.org/10.1145/3340531.3412741

considered *niche*, despite having dedicated groups of core listeners. For the less-familiar, non core-audience, search systems could surface this type of content more. This would make this content more accessible, and enable non-core listeners to discover it without alienating core-listeners.

Casual Music: Often users aspire to listen to casual music, mainly composed of non-music audio (*e.g.*, nature sounds) or instrumental tracks. Such music content is often characterized by smooth, cohesive composition and a lack of jarring or disruptive sounds. For this type of music content, users often have low or no preference on the specific content they wish to stream, and hence face trouble in articulating their information needs to access such content. This *casual* music forms the second group of content we wish to surface in search results, by identifying broad queries.

Our goal in this work is to identify queries for which a search ranker can potentially surface results from under-served content groups, such as the Niche Genres and Causal Music identified above, without hurting user satisfaction. Such a module can in turn be used by query-content matching and ranking systems to train models to surface under-served content.

Contributions. In this paper we consider the problem of identifying *non-focused queries*, *i.e.*, queries that can be targeted to boost under-served content. We draw inspiration from recent research on broad-intent queries and user mindset analysis to address the problem [19] and define *focused* and *non-focused* queries. Focused queries relate to a specific information need, *e.g.*, "*katy perry fireworks*". Non-focused queries represent a more broad and open ended information need, where users have a seed of an idea in mind and are generally more open to non-specific recommendations, *e.g.*, "*relaxing music*".

Building upon existing work on search query understanding [7, 13], we identify three classes of features derived from query characteristics and user interactions, which are predictive of identifying relevant queries to surface under-served content (Section 3). Furthermore, for each class of under-served content, we train models to predict a query level score that indicates the suitability to surface under-served content (Section 4). Due to the lack of large scale labeled data, we additionally present weak supervision approaches to learn from limited labeled data.

We present insights and findings from a large scale deployed search ranker powering a popular music streaming platform. Based on user interaction data from over 10 million users and 500K search queries, we analyze how the different query identification features perform, and present offline prediction performances across different models. Finally, we conduct a live AB test on 70 million users for a duration of one week to evaluate the predictive power of the proposed model in an online test. We contend that our findings and insights have implications on the design of multi-stakeholder search systems powering online marketplaces.

2 RELATED WORK

Search & Recommendations on Platforms. The major motivation of our work stems from the need to balance supplier preferences with user needs on multi-stakeholder platforms. Platforms and marketplaces have enjoyed a long history of detailed research [28, 29], with past work exploring competition [5], strategies [14] and economics [20] on such platforms. The concept of multiple stakeholders in recommender systems is also suggested in prior research [1]. The role of search and the need for supplier considerations is an understudied area, which we focus on in our work.

Query Understanding. To facilitate the exposure of under-served content, we rely on developing query understanding module. The problem of query understanding has enjoyed a long history of active research, with advancements around understanding query semantic [15], knowledge-based conceptualization [33], semi-supervised learning [32] and neural learning for voice query understanding on an entertainment platform [27], and query intent modeling [6, 16].

Sponsored Ads. An idea related to surfacing under-served content via search results is Sponsored Ads. Given a set of keywords, usually businesses pay for advertisements to show up in the search results when the users write a search query including those keywords. Previous works addressed this as multi-label learning problem where the words within a query are treated as labels to annotate the relevant results along with advertised content [2, 12, 26].

We build on top of such work and in particular on quantifying broadness of query aspects [30] and intent [6], and leverage such aspects to specifically consider how some queries are better suited than others to include under-served content.

3 QUANTIFYING NON-FOCUSED QUERIES

We want to detect *non-focused queries*, *i.e.*, candidate search queries for which the ranking algorithm could present under-served content to users while satisfying their search needs. An intuitive approach could be to check existing results associated with queries, and check for which queries the users are consuming under-served content. However, such content is by definition surfaced very rarely, hence the simple inspection of results for existing queries is largely ineffective. This presents a major challenge, as the direct inspection of queries and results shown and consumed by the user is non-informative for the majority of the queries. To overcome the limitation, we propose the following groups of features:

- **Standalone features:** these include surface-level information from the queries alone.
- **Reference dependent features:** these are conditioned on gold standard (reference) queries that already included under-served music content in their results, and consumed by users.
- **Interaction features:** quantifying the generality of a query.

Notation. We indicate with Q all of the queries under analysis, and Q^r indicates the reference queries, *i.e.*, those queries for which the users already are consuming under-served content. We refer to the results associated to queries as R. The results the users click on are indicated with R^c, while the results displayed (but not necessarily consumed) are indicated by R^d. The under-served content available on the platform (not necessarily included in any query result) is indicated as C.

3.1 Standalone Features

There are few queries for which users are consuming under-served content. To augment this set of queries, the first features we compute is the number of under-served content displayed to users and the number of those consumed by the users for a particular query. Formally, the number of under-served content displayed for query i, indicated as ND_i, is defined as $ND_i = |R_i^d \bigcap C|$. Similarly, the number of under-served content consumed by the users for a query i, indicated as NC_i, is defined as $NC_i = |R_i^c \bigcap C|$.

In the current system, a query is served through "Prefix Query Resolution" mechanism, hence the text is often an abridged version of the actual user intent. Therefore, the query text itself has little context for feature extraction. We assume that the clicked results, *e.g.*, track titles, album or artist names, are latent representations of the user's intent. We compute the embedding vector of a query Q as a weighted average of the `Word2Vec` [23] vectors of the clicked results. Similarly, the vector representation of a track is computed based on its co-occurrence statistics across playlists and the vector representation of an artist is the weighted average of the tracks they have performed [4].

3.2 Reference Dependent Features

Very few queries are labeled as *reference*, meaning they include under-served content consumed by users. Nevertheless, results returned for these Q^r queries can be compared against results returned for all other queries Q. We hypothesize that higher overlap among the pair of results would mean that a particular query Q_i is highly similar to a query belonging to Q^r. We compute two similarities: overlap in clicked results (S^c), and overlap in displayed results (S^d). Both are computed in terms of the number of entities overlapping divided by the size of their union, *i.e.*, the Jaccard similarity between consumed or displayed results. Formally, for any query i we compute $S_i^c = \sum_{j=1}^{|Q^r|}(J(R_i^c, Q_j^r))$, where J is the Jaccard similarity. Similarly, $S_i^d = \sum_{j=1}^{|Q^r|}(J(R_i^d, Q_j^r))$.

Embedding Distance. We conducted a pilot study and observed that, given the wide range of results available for the same query, the Jaccard similarity is 0 in most cases. To remedy this and avoid scalability problems when Q^r is large, we considering non-exact matching additional metric, the distance $d(Q, C)$, defined in terms of embeddings of the target entities R^c (artist, track, album, playlists, *i.e.*, results that users click on for the particular query), with respect to under-served content C. Formally, $d(Q, C) = f(d_1, \ldots, d_{|C|})$, where f computes the average of its 5 smallest inputs, and $d_i(Q, C_i) = L2\_distance(avg(R^c), C_i)$. The distance is computed using FAISS, a fast approximate nearest neighbor algorithm [17]. The distance is useful when there is currently no under-served content displayed. Queries that exhibit a low $d(Q, C)$ while not including any of such content include *study music*, *peace*, and *sleep stories*.

Pronunciation distance (Dist Prons). . This metric measures the weighted pronunciation distance between the reference queries Q^R and rest of the queries. The pronunciation distance metric is a customized Levenshtein distance [18] that overlooks the orthographic differences to capture the similarity between the query texts. First we convert the query text into sequence of phonemes by applying Grapheme-to-Phoneme (G2P) model trained on a recurrent neural network (RNN) with long short-term memory units (LSTM) architecture [24]. Since queries tend to be very noisy, it is advisable to ignore commonly confused pairs of phonemes (*e.g.*, {*D, DH, T, TH*}). To this end, we choose a lower edit cost for commonly confused pairs while computing the distance, otherwise all edit costs are 1 and the *lower* edit costs are derived from the statistics discussed in [3]. Then we compute the distance between pairs of Q and Q^R phoneme sequences. An advantage of pronunciation distance over the lexical edit distance is the ability to detect common misspellings, word elongations, or incomplete strings of a reference query. The table below shows that the pronunciation distance is 0 for the pair of incorrect and correct query ("*Randy Rhoads*" pronounced as "R AE N D IY R OW D Z") which means they are highly similar, whereas the lexical distances are non-zero values.

Wrong spellings	Lexical distance	Phonetic spelling	pronunciation distance
Randy Roads	1	R AE N D IY R OW D Z	0
Rhandy Rohads	3	R AE N D IY R OW D Z	0
Rhaandhy Rhoadzzz	6	R AE N D IY R OW D Z	0

Knowledge Graph Distance (Dist Wiki KG). Queries that often share similar results tend to share ontological roots. To capture this, we linked the queries to entities on Wikipedia Knowledge Graph (KG) using an open-source entity linking toolkit, Fast Entity Linker [8, 25], mapping the partial queries to canonical KG entities. Then, we measure the distance between the embeddings of KG entities (corresponding to their respective queries) using the pre-trained embeddings model *Wikipedia2Vec* [34]. Query text mapped to their respective KG entities could disambiguate queries that are lexically similar but refer to different entities and are distant in the embedding space. The example below shows queries with a common phrase "small town" but referring to different KG entities:

Query text	Wikipedia KG entity
small town usa	Small_Town_USA
small town girl	Small_Town_Girl_(song)
break up in a *small town*	Break_Up_in_a_Small_Town
small town saturday night	Small_Town_Saturday_Night_(song)

3.3 Interaction based Features

Click Entropy. This indicates whether a query is highly *non-focused* or not. For a query q, the entropy H_q is computed as $H_q = -\sum_k (p(R_{q,k}^c) * \log(p(R_{q,k}^c)))$, where $p(R_{q,k}^c)$ is the probability of the result $R_{q,k}^c$ to be consumed by the users. As this value is not analytically computable, we estimate it by counting the number of times that users clicked on a particular result based on the same query. Entropy indicates broad intent understanding. Simultaneously, there is a strong correlation between an unfocused query and receptiveness of a user to explore novel content [31]. Examples of queries that include under-served content and have high entropy are *wedding*, *instrumental*, *sad*, *morning*.

4 PREDICTION OF NON-FOCUSED QUERIES

The proposed features carry information on which non-focused queries are better suited to return under-served content. Based on these, we want to learn a model to predict whether an unlabeled query could help presenting under-served content in the search results. We devised two manual thresholding-based predictors, and three machine learning models using the proposed features.

4.1 Feature Thresholds as Predictors

First, we analyzed threshold-based predictors. For each feature under analysis, the predictor aim at predicting the output based on an evolving threshold. For threshold ρ_s and a feature s, we regarded the examples having the feature under examination higher than the threshold as positive, and those with the feature lower than the threshold as negative. Formally, the prediction $y_{ts} = 1 \iff s \geq \rho_s$, $y_{ts} = 0$ otherwise, where s is a non-distance-based feature. For distance-based features the prediction is reversed, because of the negative correlation of distance-based features with the output to predict. ρ_s is a feature specific threshold that has been cross-validated on a learning set X_{lr} after multiple splits in training and validation (10 fold cross-validation).

We also tested a pairwise combination between the features, based on a pair of features and thresholds using the logical AND between two features, as: $y_{ts} = 1 \iff s_1 < \rho_1$ AND $s_2 < \rho_2$, $y_{ts} = 0$ otherwise. As before, when s_1 or s_2 are distance-based features, we consider $s_1 > \rho_1$ and/or $s_2 > \rho_2$.

We then employed an automated procedure to find the best multiple combination of features, using a decision tree classifier [11] to automatically devise a prediction rule to identify which queries could potentially include under-served content. A decision tree classifier builds a "decision tree", meaning that starting from the most discriminative feature, it employs a set of threshold-based rules on the features to arrive at the best prediction. As this procedure is prone to overfitting [9], we employ a grid search cross-validation procedure limited to height tree structure of 5 to find the best decision tree. Not only do decision trees provide an automated way to infer the best thresholds, they also allow us to easily interpret the results through their inspection.

Each sample is associated with a probability of belonging to one class. Based on this, we consider two ways to assign the class label to the sample: we assign a class if the probability of the label is higher than 0.5, or we change the probability threshold after cross-validating the threshold on the validation set.

4.2 Trained Models

Threshold-based predictors have clear limitations, such as failing to learn non-linear relationships. To overcome this limitation, we use two statistical models trained on the features discussed above. First we trained a random forest classifier, an ensemble learning method that constructs a multitude of decision trees during training and returns the class that corresponds to the mode of the classes of the individual trees during prediction [10]. Due to random feature selection and bagging, random forests are known to alleviate overfitting problems. We train and validate the model using a 10-fold cross validation procedure for optimal parameters. We also use a neural network architecture comprising two dense layers (with 128 and 64 neurons, respectively) with a dropout of 0.5 between the layers, which we train for 100 epochs.

4.2.1 Leveraging Weak Supervision. Query labeling has a high cost. The majority of queries are unlabeled, hence we prefer to use the few labeled queries to improve the predictive power of our models. We employ weak supervision techniques with inaccurate labeling [35]. To assign a weak supervision label to unlabeled queries, we explored several approaches. First, we experimented on which model to use to assign the weak supervision label, between the decision tree and the random forest classifier, and we chose the latter as it performed best on the validation set. Then, we estimated the output labels of all queries Q using the random forest classifier, and regarded such labels as *weak supervision labels*. Second, we considered different ways to use the weak supervision labels: (i) using only a subset of the weakly-labeled queries, based on the probability of label assignment, *i.e.*, only considering the queries for which the model assigned a label with probability higher than 0.8, or (ii) considering all 500K queries with the weak supervision label, but using the probability of label assignment as sample weight during the training procedure of the neural network. We empirically selected the latter option, based on an higher performance on the validation set.

4.3 Ensemble

Each classifier, as empirically shown in Section 5, has its own strength and weakness. To benefit from all of the different classifiers, we devised an ensemble model based on the predictions of the best performing classifiers, namely the threshold-based (using *Entropy*, *Pronunciation distance*, *Min Dist Content*, *Dist Reference Queries*) and the neural network classifier. The ensemble model is a voting mechanism with three different voting criteria: $En(q) = 0$ if at least one classifier identifies non-focused query, 1 if at least two classifiers agree, or 2 if at least three classifiers agree. Such voting criterion aims at higher recall: when any of the classifiers predict a query to be non-focused, then there is potential to present under-served content to the user.

5 EXPERIMENTS & RESULTS

We conduct three empirical evaluations to investigate the effectiveness of different techniques to identify queries to surface under-served music content. First, we gather labeled test data from music domain experts and evaluate the models on it. Then, we conduct gain vs. loss based comparison to showcase the trade-off between surfacing more under-served content and drop in relevance. Finally, we demonstrate performance gains via a live A/B test in Section 6.

Dataset Description. We use logged feedback data and live production traffic from an online music streaming service to understand how users search for music content. We logged the search result pages returned for each query along with users' interactions (such as clicks or taps) on the results. This dataset consists of ~500K unique queries, extracted from a random sample of 35M queries in May 2020. The queries are selected if they include more than two characters. Also, we limit queries by only considering US users. Starting from the original 35M queries, those have been grouped together by the query text (exact matching after trimming and clustering of queries having at least the first three letters in common, based on the number of entities), which resulted in ~500K queries.

Using a stratified sampling across the query population with entropy and distance features, we selected 1000 queries for manual annotation by a team of in-house subject matter experts in music content and culture. The annotation task was performed separately for the *Niche Genre* and *Casual Music* groups. For each query, an

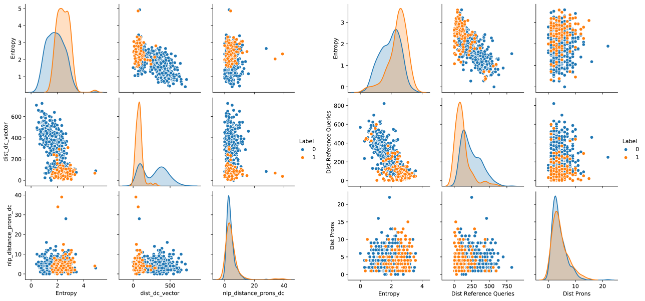

Figure 1: Pairplot between the most discriminative features on Casual Music (left) and on Niche Genres (right). The distributions are plotted on the labeled dataset. The label indicates whether a query could potentially include casual music / niche genres.

expert was asked to label whether we could surface search results from those groups or not. We regard these queries as the annotated dataset, used to evaluate different models. We use the other 500K queries as weak supervision dataset to train models at scale.

5.1 Evaluating Threshold based Predictors

We investigate different predictors on the expert annotated labels by looking at the predictive value of features.

5.1.1 Feature Analysis. Figure 1 shows the distribution of some of the features discussed above on a dataset of labeled queries, *i.e.*, for which we have the labels associated to the queries. We indicate with `class 0` the focused queries and with `class 1` the non-focused queries, *i.e.*, candidates to support under-served content. The left plot represents the analysis on Casual Music, while the right plot represents the analysis on Niche Genre. Notably, especially for the left plot, *Dist Reference Queries* (embedding distance from reference queries) almost displays a bi-modal distribution that discriminates between the target classes. Feature values that best identify `class 1` are concentrated on the left part of the plot. This means that a single threshold on this feature could potentially discriminate the majority of the queries. On the other hand, *Entropy* and *Pronunciation Distance* are evenly distributed, thus moderately discriminating the target classes. When observed in combination, the features achieve higher discrimination power as shown in the sub-figure of *Embedding Distance & Entropy*, as well as *Embedding Distance & Pronunciation Distance*. This means that the features could be indeed beneficial for the discrimination task, as discussed later.

Generally, we note how such features are less separated in the Niche Genre case. This suggests that the query classes on such task may be harder to separate. We show later (Section 5.1.2) how this is indeed the case, as overall the results for the models trained on predicting the classes of queries for Casual Music perform generally better than those trained to predict for Niche Genre.

Table 1 includes the Pearson correlations between the single features (sorted by decreasing absolute value) for Casual Music (top part of the table) and Niche Genre (bottom part). First, we note how the relative importance of the features changes across the two groups, suggesting that the two tasks need to be considered separately, as the predictors would need to use the same features in a different manner. For Niche Genre, we observe that the correlations are higher for Jaccard similarity for the displayed (and clicked) results between queries Q and reference queries Q^R. This is an expected correlation, mainly because the user's intent is captured in the results and higher overlap in both sets of results indicate better predictability of the target label.

		ρ
Casual Music	Embedding distance $d(Q, Q^r)$	-0.454
	Click entropy of a query H_Q	0.332
	Jaccard for displayed (S_i^d)	0.306
	Jaccard for clicked (S_i^c)	0.248
	No. displayed results $\lvert R^d \rvert$	0.191
	No. clicked results $\lvert R^c \rvert$	0.169
	Knowledge graph distance	-0.056
	Pronunciation distance	0.040
	Embedding distance to content $d(Q, C)$	-0.018
Niche Genre	Jaccard for displayed (S_i^d)	0.407
	Jaccard for clicked (S_i^c)	0.405
	Embedding distance $d(Q, Q^r)$	-0.369
	Embedding distance to content $d(Q, C)$	-0.305
	Click entropy of a query H_Q	0.291
	No. displayed results $\lvert R^d \rvert$	0.232
	No. clicked results $\lvert R^c \rvert$	0.185
	Pronunciation distance	0.060
	Knowledge graph distance	-0.005

Table 1: Correlation with label of single features (sorted by absolute value) for Casual Music and Niche Genres.

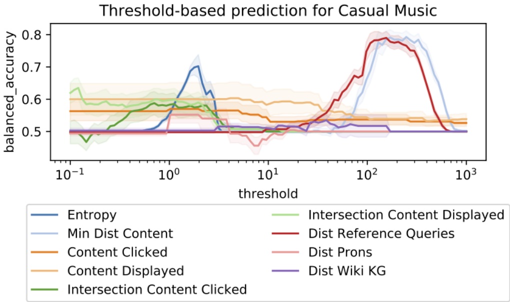

Figure 2: Prediction results using a single threshold-based predictors on Casual Music.

For Casual Music, the highest (in absolute value) correlations are for the *embedding distance* to the reference queries, meaning that there are high similarities on queries associated with Casual Music, and click entropy, which suggests how users, when searching for casual music, do not have a particular track in mind. Indeed, the entropy of a query is well correlated with under-served content due to the inherent nature of unfocused queries: non-focused queries lead to an higher chance for the user to consume a wider range of results proposed by the search system. Combination of the features yield even better correlation with the target label and this is intuitive based on the feature value distribution in Figure 1.

5.1.2 Predictive Performance. Figure 2 shows how the prediction based on single feature thresholds perform on Casual Music. Similar considerations could be done also for Niche Genres. Detecting the right threshold to use is important, and greatly affects performance.

We perform a grid search on the threshold values on the validation dataset, across all features and plot the corresponding balanced accuracy result. We observe that as indicated by the correlational analysis in Section 3, thresholding on the distance with respect to other non-focused queries gives the best prediction accuracy. Entropy based predictor gives accuracy of 0.70 with entropy threshold of 1.96, which indeed highlights that queries with higher click entropy are non-focused queries.

5.2 Performance of Learnt Models

Beyond predictors based on feature thresholds, we can also train models based on the different query features identified. We use three supervised models: a simple decision tree classifier, a random forest classifier and a neural network classifier. Table 2 presents the predictive performance for different methods for *Casual Music* and *Niche Genre* content groups. We report precision, recall and F1-score for both classes (0: candidate query unsuitable for under-served content, 1: candidate query suitable to include more under-served content) to highlight different properties of the features. In particular, whether the goal is to increase the number of queries in which to include more under-served content, a system designer would prefer an high recall for `class 1`. The trade-off is in particular with respect to the precision for `class 0`, as a drop in precision for 0 may result in under-served content surfaced in non-suitable queries, potentially resulting in a loss of user satisfaction.

In both groups, distance-based features (Dist Reference Queries and Min Dist Content) perform particularly well in terms of recall for `class 1`, as they only consider distance in terms of the embeddings of the results. Instead, because of the limit of the currently displayed under-served content, "Content Clicked" and "Content Displayed" have almost the worst recall for `class 1`.

Trained models give best accuracy and overall a better balance between the performance on both classes. Notably, for Casual Music, the best accuracy is achieved by the random forest classifier, which is overall the best predictor. However, by looking at the recall for `class 1`, this model is missing out on many queries that can include under-served content, with a recall of just 0.226, almost the worst across all classifiers. Instead, the best recall is given by Ensemble 0, which is expected as a query is deemed a candidate for under-served content if any of the classifiers in the voting return `class 1` for the query. The recall for `class 1` of Ensemble 1 and 2 are lower, but still among the highest for Casual Music group. Interestingly the neural network model could not outperform the random forest. However, it contributes to the voting process in the Ensemble classifier that achieves best recall for the `class 1` in both music content groups. We argue that this is an important result since it directly impacts the presentation of under-served content, as higher recall for `class 1` is more desirable than higher overall accuracy.

5.2.1 Impact of Weak Supervision. We use the model learnt on the curators labels to label all queries in the extended dataset (500k queries), out of which only 650 had a manually assigned label. We regarded the predicted labels as *weak supervision labels*, using those to fine-tune a neural network which was trained on the domain expert labeled dataset. We run this additional training for 500 epochs. Then, we fine-tuned the network further on the initial labeled training data, to ensure that the final weights would be tuned with the strongly-assigned labels. From Table 2, the overall accuracy between the neural network classifier and the neural network classifier after weak supervision is decreasing. However, by closer inspection, we can make two important considerations. First, for both music content groups, the precision of the `class 0` is increasing. This means that overall the network becomes more aware on the queries that are not suitable to include additional under-served content. Next, we see a huge impact on the recall for `class 1`, especially for Casual Music group (from 0.758 to 0.903). After the weak supervision, the network has improved in not missing out on the queries suitable for serving casual music and niche genres. Despite differences in content across *Niche Genre* and *Casual Music*, we observe similar trends in performance across both content groups. We can conclude that the weak supervision labels bring value to the learning models, in better identifying queries that can potentially include under-served content.

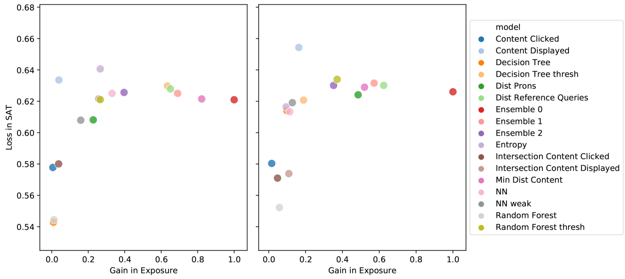

Figure 3: Gain in exposure vs Loss in satisfaction results from live A/B test, for *Casual Music* content group (Left) and *Niche Genres* content group (Right).

6 LIVE A/B TEST FOR SURFACING CONTENT

Identifying *non-focused* queries provides platforms with the opportunity to surface under-served content to the users – content that otherwise users may not discover. System designers often have to trade-off between balancing user satisfaction and supplier exposure on the platform. Surfacing more under-served content helps exposing undiscovered content, but can also affect user satisfaction when irrelevant content gets ranked higher on search results.

To study the effectiveness of our proposed approach to surface under-served content we conducted a live A/B test on our production search platform. In addition to the default ranker, we trained two alternative rankers by using features discussed in the previous sections that indicate whether a candidate result belongs to *Niche Genre* content group or not, and to *Casual Music* group or not. We then boost weights of those features relative to other default features in the ranker. The new rankers are expected to increase the exposure of results from the two content groups. There are potentially other strategies to surface under-served content, which is beyond the scope of this paper.

We conducted a week long A/B test wherein users were randomly divided into three test cells, assigning the default ranker to *control* group, and assigning the new rankers that surfaces *Niche Genre* content and *Casual Music* content to the two *treatment* groups. We collected a large sample of interaction data covering 70 million

		class 0			class 1			
		precision	recall	f1-score	precision	recall	f1-score	accuracy
Single Feature Thresholding	Content Clicked	0.837	0.982	0.904	0.643	0.145	0.237	0.829
	Content Displayed	0.848	0.964	0.902	0.583	0.226	0.326	0.829
	Dist Reference Queries	0.984	0.669	0.797	0.391	0.952	0.554	0.721
	Dist Prons	0.812	0.856	0.834	0.149	0.113	0.128	0.721
	Dist Wiki KG	0.824	0.856	0.840	0.216	0.177	0.195	0.732
	Entropy	0.934	0.561	0.701	0.295	0.823	0.434	0.609
	Intersection Content Clicked	0.840	0.924	0.880	0.382	0.210	0.271	0.794
	Intersection Content Displayed	0.863	0.838	0.850	0.357	0.403	0.379	0.759
	Min Dist Content	0.989	0.629	0.769	0.368	0.968	0.533	0.691
Combination Thresholding	Dist Prons AND Dist Reference Queries	0.979	0.673	0.797	0.389	0.935	0.550	0.721
	Entropy AND Dist Reference Queries	0.985	0.691	0.812	0.407	0.952	0.570	0.738
	Entropy AND Dist Prons	0.917	0.594	0.721	0.294	0.758	0.423	0.624
	Entropy AND Intersection Content Displayed	0.870	0.917	0.893	0.511	0.387	0.440	0.821
	Entropy AND Intersection Content Clicked	0.861	0.845	0.853	0.358	0.387	0.372	0.762
	Intersection Content Displayed AND Dist Referen...	0.876	0.942	0.908	0.610	0.403	0.485	0.844
	Intersection Content Displayed AND Dist Prons	0.863	0.838	0.850	0.357	0.403	0.379	0.759
	Intersection Content Clicked AND Dist Reference...	0.880	0.921	0.900	0.551	0.435	0.486	0.832
	Intersection Content Clicked AND Dist Prons	0.845	0.964	0.901	0.565	0.210	0.306	0.826
Trained Models	Decision Tree	0.832	**1.000**	0.908	**1.000**	0.097	0.176	0.835
	Decision Tree after Threshold	0.984	0.673	0.799	0.393	0.952	0.557	0.724
	Random Forest	0.853	**1.000**	**0.921**	**1.000**	0.226	0.368	**0.859**
	Random Forest after Threshold	0.971	0.723	0.829	0.421	0.903	**0.574**	0.756
	NN	0.932	0.741	0.826	0.395	0.758	0.519	0.744
	NN weak supervision	0.970	0.705	0.817	0.406	0.903	0.560	0.741
	Ensemble 0	**1.000**	0.327	0.493	0.249	**1.000**	0.399	0.450
	Ensemble 1	0.988	0.583	0.733	0.341	0.968	0.504	0.653
	Ensemble 2	0.984	0.673	0.799	0.393	0.952	0.557	0.724
Single Feature Thresholding	Content Clicked	0.826	0.989	0.900	0.842	0.219	0.348	0.827
	Content Displayed	0.828	0.982	0.898	0.773	0.233	0.358	0.824
	Dist Reference Queries	0.919	0.704	0.798	0.409	0.767	0.533	0.718
	Dist Prons	0.763	0.646	0.700	0.157	0.247	0.191	0.562
	Dist Wiki KG	0.789	**0.996**	0.881	0.000	0.000	0.000	0.787
	Entropy	0.889	0.701	0.784	0.374	0.671	0.480	0.695
	Intersection Content Clicked	0.842	0.953	0.894	0.649	0.329	0.436	0.821
	Intersection Content Displayed	0.852	0.945	0.896	0.651	0.384	0.483	0.827
	Min Dist Content	0.872	0.774	0.820	0.404	0.575	0.475	0.732
Combination Thresholding	Dist Prons AND Dist Reference Queries	0.924	0.668	0.775	0.389	0.795	0.523	0.695
	Entropy AND Dist Reference Queries	**0.929**	0.668	0.777	0.393	0.808	0.529	0.697
	Entropy AND Dist Prons	0.867	0.737	0.797	0.368	0.575	0.449	0.703
	Entropy AND Intersection Content Displayed	0.852	0.945	0.896	0.651	0.384	0.483	0.827
	Entropy AND Intersection Content Clicked	0.835	0.978	0.901	0.769	0.274	0.404	0.830
	Intersection Content Displayed AND Dist Referen...	0.852	0.949	0.898	0.667	0.384	0.487	0.830
	Intersection Content Displayed AND Dist Prons	0.849	0.945	0.895	0.643	0.370	0.470	0.824
	Intersection Content Clicked AND Dist Reference...	0.843	0.964	0.899	0.706	0.329	0.449	0.830
	Intersection Content Clicked AND Dist Prons	0.835	0.978	0.901	0.769	0.274	0.404	0.830
Trained Models	Decision Tree	0.839	0.967	0.898	0.710	0.301	0.423	0.827
	Decision Tree after Threshold	0.865	0.938	0.900	0.660	0.452	0.537	0.836
	Random Forest	0.851	0.982	**0.912**	**0.839**	0.356	0.500	**0.850**
	Random Forest after Threshold	0.904	0.788	0.842	0.463	0.685	**0.552**	0.767
	NN	0.850	0.971	0.906	0.765	0.356	0.486	0.841
	NN weak supervision	0.853	0.956	0.902	0.700	0.384	0.496	0.836
	Ensemble 0	0.922	0.391	0.549	0.277	**0.877**	0.421	0.493
	Ensemble 1	0.920	0.668	0.774	0.385	0.781	0.516	0.692
	Ensemble 2	0.875	0.814	0.843	0.446	0.562	0.497	0.761

Table 2: Accuracy, Precision, Recall, F1 score for both classes of queries: Casual Music (top half) and Niche Genre (bottom half).

users, interacting with 86 million results, in 600 million sessions and 56 million queries. One lens to interpret these results is in terms of the *control* group that uses the default ranker optimized to balance discovery of new content against potentially irrelevant content. Ideally, the optimal method should increase the number of queries for which we were able to surface results from under-served content group, while minimizing user dissatisfaction.

We identify queries for which the *treatment* ranker surfaced a search result from *Niche Genres* and *Casual Music*, and the control ranker did not. We also logged whether the user interacted with the newly surfaced result from the *Niche Genre* content group, and used that information to assign satisfaction label to the query. If the user streamed music from the surfaced content we assign a label of +1 to the query, 0 otherwise, and we checked the agreement with our classifiers. For each method, we compute two metrics: *(i) Gain in Exposure*, computed based on the number of *non-focused* queries identified by the method. Higher numbers are preferred, since higher number of such identified queries will lead to greater opportunity to surface content from *Niche Genre* content group. Note that models optimizing solely for this might end up over-surfacing such results, which might lead to user dis-satisfaction. *(ii) Loss in Satisfaction*, computed using the proportion of queries where users did not stream the surfaced under-served content for each query tagged as *non-focused* query by any method. Lower numbers are preferred, as we prefer the model to minimize polluting results by unnecessarily exposing irrelevant content to users.

An ideal method would maximize the **Gain in Exposure** while minimizing the **Loss in Satisfaction** metric. Figure 3 presents the results comparing the different methods on these two metrics, for both content groups. We observe a good spread of the different models across the scatter plot, which indicates that good trade-off based decisions can be made here. This empowers system designers to select solutions on a need-basis, depending on platform's current business requirements. Some methods (*e.g.*, Entropy) fare poorly in giving useful trade-off for *Niche Genre* group, since they suffer from loss in satisfaction without offering much gain in exposure. On the other hand, some methods are fairly conservative (*e.g.*, Decision tree classifier and Intersection of Displayed Content), which help platforms not to risk any loss in satisfaction but reducing the number of exposures. Ensemble-0 provides the best trade-off: significant gains in exposure, with comparable loss in satisfaction.

The Live A/B test highlights that the problem of exposing under-served content is one of making trade-off based decisions. Certain models do offer the advantage of giving more exposure to under-served content, while minimizing the loss of user satisfaction; it is the choice of the system designer to select which methods are desired when, based on platform's business needs.

7 CONCLUSION

In this work we proposed multiple features to identify which queries are better suited to return under-served content while maintaining user satisfaction. The proposed features are easily computable for a large set of queries, independently from the fact that they may not already relate to under-served content. Our results show the efficacy to use the proposed features in conjunction with a wide range of models. Paired with the learning models, the prediction using the features achieves high accuracy, and the detailed results on both classes of suitable / non-suitable queries for under-served content show how each model benefit differently from the features. The results on both classes of queries highlight the importance of understanding the benefit of each classifier, as this needs to be taken into consideration by a system designer. Identifying the right queries to include additional under-served content is a fundamental step to ensure a healthy, sustainable marketplace.

REFERENCES

[1] H. Abdollahpouri, R. Burke, and B. Mobasher. [n.d.]. Recommender Systems as Multistakeholder Environments. In *Proceedings of UMAP 2017*.

[2] Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. 2013. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of WWW 2013*. 13–24.

[3] Rimah Amami and Noureddine Ellouze. 2015. Study of phonemes confusions in hierarchical automatic phoneme recognition system. *arXiv preprint arXiv:1508.01718* (2015).

[4] Ashton Anderson, Lucas Maystre, Ian Anderson, Rishabh Mehrotra, and Mounia Lalmas. 2020. Algorithmic effects on the diversity of consumption on spotify. In *Proceedings of The Web Conference 2020*. 2155–2165.

[5] Mark Armstrong. 2006. Competition in two-sided markets. *The RAND Journal of Economics* 37, 3 (2006), 668–691.

[6] Azin Ashkan, Charles LA Clarke, Eugene Agichtein, and Qi Guo. 2009. Classifying and characterizing query intent. In *European conference on information retrieval*. Springer, 578–586.

[7] Ricardo Baeza-Yates. 2017. Semantic query understanding. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1357–1357.

[8] Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. 2015. Fast and space-efficient entity linking for queries. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. 179–188.

[9] Max Bramer. 2013. *Avoiding Overfitting of Decision Trees*. Springer London, London, 121–136. https://doi.org/10.1007/978-1-4471-4884-5_9

[10] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[11] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. 1984. Classification and regression trees. Belmont, CA: Wadsworth. *International Group* 432 (1984), 151–166.

[12] Sheng Chen, Akshay Soni, Aasish Pappu, and Yashar Mehdad. 2017. DocTag2Vec: An Embedding Based Multi-label Learning Approach for Document Tagging. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*. 111–120.

[13] W Bruce Croft, Michael Bendersky, Hang Li, and Gu Xu. 2011. Query representation and understanding workshop. In *ACM SIGIR Forum*, Vol. 44. ACM New York, NY, USA, 48–53.

[14] Thomas Eisenmann, Geoffrey Parker, and Marshall W Van Alstyne. 2006. Strategies for two-sided markets. *Harvard business review* 84, 10 (2006), 92.

[15] Homa B Hashemi, Amir Asiaee, and Reiner Kraft. 2016. Query intent detection using convolutional neural networks. In *International Conference on Web Search and Data Mining, Workshop on Query Understanding*.

[16] Jian Hu, Gang Wang, Fred Lochovsky, Jian-tao Sun, and Zheng Chen. 2009. Understanding user's query intent with wikipedia. In *Proceedings of the 18th international conference on World wide web*. 471–480.

[17] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. *arXiv preprint arXiv:1702.08734* (2017).

[18] Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, Vol. 10. 707–710.

[19] Ang Li, Jennifer Thom, Praveen Chandar, Christine Hosey, Brian St Thomas, and Jean Garcia-Gathright. 2019. Search mindsets: Understanding focused and non-focused information seeking in music search. In *WWW*. 2971–2977.

[20] Bertin Martens. 2016. An economic policy perspective on online platforms. (2016).

[21] Rishabh Mehrotra, Mounia Lalmas, Doug Kenney, Thomas Lim-Meng, and Golli Hashemian. 2019. Jointly leveraging intent and interaction signals to predict user satisfaction with slate recommendations. In *WWW*. 1256–1267.

[22] Rishabh Mehrotra, James McInerney, Hugues Bouchard, Mounia Lalmas, and Fernando Diaz. 2018. Towards a fair marketplace: Counterfactual evaluation of the trade-off between relevance, fairness & satisfaction in recommendation systems. In *Proceedings of the 27th acm international conference on information and knowledge management*. 2243–2251.

[23] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[24] Benjamin Milde, Christoph Schmidt, and Joachim Köhler. 2017. Multitask Sequence-to-Sequence Models for Grapheme-to-Phoneme Conversion.. In *INTERSPEECH*. 2536–2540.

[25] Aasish Pappu, Roi Blanco, Yashar Mehdad, Amanda Stent, and Kapil Thadani. 2017. Lightweight multilingual entity extraction and linking. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. 365–374.

[26] Yashoteja Prabhu, Anil Kag, Shrutendra Harsola, Rahul Agrawal, and Manik Varma. 2018. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In *WWW 2018*. 993–1002.

[27] Jinfeng Rao, Ferhan Ture, and Jimmy Lin. 2018. Multi-task learning with neural networks for voice query understanding on an entertainment platform. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 636–645.

[28] Marc Rysman. 2009. The economics of two-sided markets. *Journal of Economic Perspectives* 23, 3 (2009), 125–43.

[29] Srinivasaraghavan Sriram, Puneet Manchanda, and Bravo. 2015. Platforms: a multiplicity of research opportunities. *Marketing Letters* (2015).

[30] Xuanhui Wang, Deepayan Chakrabarti, and Kunal Punera. 2009. Mining broad latent query aspects from search sessions. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 867–876.

[31] Yu Wang and Eugene Agichtein. 2010. Query Ambiguity Revisited: Clickthrough Measures for Distinguishing Informational and Ambiguous Queries. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 2-4, 2010, Los Angeles, California, USA*. The Association for Computational Linguistics, 361–364.

[32] Ye-Yi Wang, Raphael Hoffmann, Xiao Li, and Jakub Szymanski. 2009. Semi-supervised learning of semantic classes for query understanding: from the web and for the web. In *Proceedings of the 18th ACM conference on Information and knowledge management*. 37–46.

[33] Zhongyuan Wang, Kejun Zhao, Haixun Wang, Xiaofeng Meng, and Ji-Rong Wen. 2015. Query understanding through knowledge-based conceptualization. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[34] Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia. *arXiv preprint 1812.06280v3* (2020).

[35] Zhi-Hua Zhou. 2018. A brief introduction to weakly supervised learning. *National Science Review* 5, 1 (2018), 44–53.