Topic Recommendation for Call Centers

Francisco Andrade, José Manuel Domínguez & Felipe Pattillo

Abstract

In this paper, we approach the problem of recommendations for call centers. More specifically we use data mining techniques such as Latent Dirichlet Allocation (LDA), Latent Semantic Indexing (LSI) and Hierarchical Dirichlet Process (HDP) to extract relevant topics discussed in each call. We assess the likelihood of these topics leading to a successful call so we can later recommend these topics i.e. the most representative words of the topic for future use. We analyzed calls made by one company, the calls made are all in spanish and we use common industry metrics to evaluate the success of a call. We divided the calls according to the type of businesses that were contacted, we use these business types as our users, the topics as our items to recommend and the success score as the rating. Finally, we make recommendations using two basic algorithms: *Slope One* and *Funk's SVD* the results show an improvement over most-popular and *success percentage.* Further improvements can be made given a larger dataset and more modern algorithms.

Context

The problem arises from the call center industry. In the year 2015 this industry generated 224.003 million pesos in revenue in Chile. This meant a 12.5% increase with relation to the previous year. With respect to the contact channel in call center type services 84,7% corresponded to voice, 6,4% face-to-face, 5,4% by email and the rest chat, Internet, social media,text messages and others.[3].

With this in mind an entrepreneurship named Midete is born. Midete attempts to
analize sales calls and deliver different metrics and statistics with the objective of improving the salesman and increasing their sales. Currently Midete has the calls transcription, metrics such as time spoken by the salesman, time spoken by client, longest monologue time (uninterrupted speech) of both the salesman and the client. We wanted to evaluate the possibility of using recommender systems to help them further improve the sales calls for a better chance of closing a deal.

State of the Art

Previously, recommender systems have been implemented in call centers but they have focussed on recommending the salesman which client to call and the product they are more likely to buy [1]. Other studies have focused on recommending procedures or solutions to the call operary for quicker assistance to the client, they did this by analyzing what was being spoken so the operary doesn't lose time searching the problem and it's solution [2].
For the stage of topic extraction, Padmanabhan & Kummamuru propose a similar approach to the process described in this paper. However, they propose topic identification, followed by a separation of the agent's (salesperson's) sentences from the customer's sentences for analysis (whereas we only focus on the salesperson's side). In addition, they propose additional SPTS clustering which is based on each sub-step of the call, meaning each procedure from the call is identified. [6]

Something important to note is that a large percentage of the related work is centered around topic extraction through data mining, with the goal of studying the results to make improvements. The recommendation process we describe involves a field that hasn't been thoroughly explored.

Proposed Solution

For the solution, we start from the basis of assuming that different types of clients are interested in different aspects of the product or have different requirements and reasons for wanting the product. This is not a far fetched assumption although all the calls made are offering basically the same product (varying the plan offered) different business types (clients) might need different requirements. We also think that the speech that the salesperson uses while contacting a beauty salon is different to the one used while contacting medical clinics, not only the technical speech varies (words used exclusively on that business) but also other aspects may vary like the problem that is being solved. We can assume this because in some of the metadata the "dolor principal" or main pain is listed and by inspecting it we found that there are subtle changes between one type of business and another. We hypothesize that identifying these subtle changes might be key at the time of closing a deal.

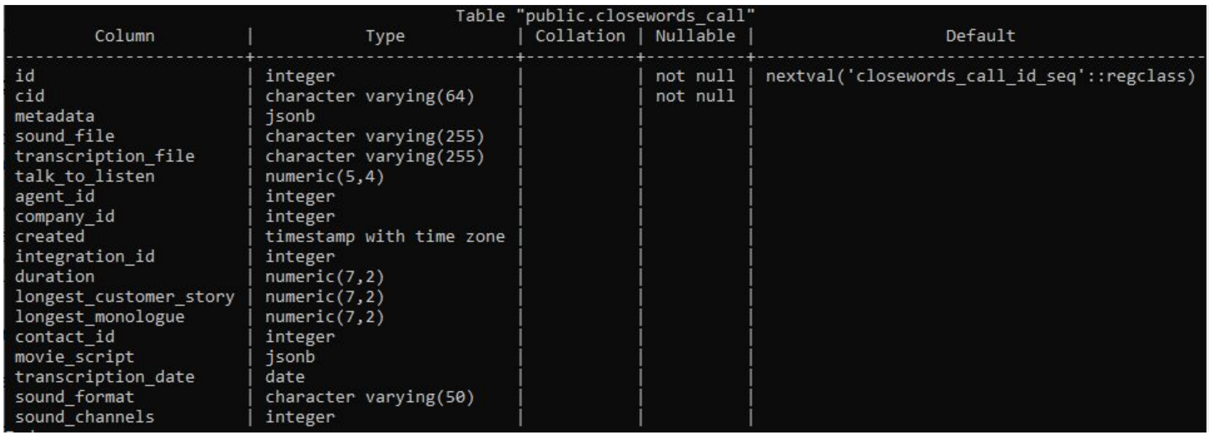
We attempt to use recommender systems with the purpose of recommending to the salesman the best topic (list of words) to use given the business type that is being contacted. For this we divided the dataset in users (business types), items (topics) and rating that is a score calculated with different metrics of the call, a detailed explanation on how this is achieved is provided in "Methodology" section. This information can later be imputed to common recommender algorithms to generate recommendations.

We use data mining techniques such as Latent Dirichlet Allocation (LDA), Latent Semantic Indexing (LSI) and Hierarchical Dirichlet Process (HDP) to extract relevant topics discussed in each call. More modern algorithms could be used in further studies such as BERT (the spanish variation BETO) for better results.

In this paper, we use Funk's SVD and slope one algorithms and compare them with the most popular and *success percentage*. Although these algorithms are not state of the art in terms of recommendations we aim to study the feasibility of using recommendation algorithms for this purpose and present several ways in which this can be done.

Dataset

Fig 4.1



```
                                        Table "public.closewords_call"
         Column          |           Type           | Collation | Nullable |                   Default
-------------------------+--------------------------+-----------+----------+---------------------------------------------
 id                      | integer                  |           | not null | nextval('closewords_call_id_seq'::regclass)
 cid                     | character varying(64)    |           | not null |
 metadata                | jsonb                    |           |          |
 sound_file              | character varying(255)   |           |          |
 transcription_file      | character varying(255)   |           |          |
 talk_to_listen          | numeric(5,4)             |           |          |
 agent_id                | integer                  |           |          |
 company_id              | integer                  |           |          |
 created                 | timestamp with time zone |           |          |
 integration_id          | integer                  |           |          |
 duration                | numeric(7,2)             |           |          |
 longest_customer_story  | numeric(7,2)             |           |          |
 longest_monologue       | numeric(7,2)             |           |          |
 contact_id              | integer                  |           |          |
 movie_script            | jsonb                    |           |          |
 transcription_date      | date                     |           |          |
 sound_format            | character varying(50)    |           |          |
 sound_channels          | integer                  |           |          |
```

Table 4.1

Field Name	Description
ID	Call identifier number
Duration	Call duration in milliseconds
movie script	JSON of the call transcription, in dialog form it alternates between what was said by the customer and the salesperson.
longest customer story	Maximum duration in seconds where the customer talk uninterrupted
longest monologue	Maximum duration in seconds where the salesperson talk uninterrupted
talk to listen	Ratio of time spoken by the salesperson and time spoken by the customer (0 if the salesperson spoke 100% of the time; 1 if the customer spoke 100% of the time)
metadata	Some Data surrounding the call, from here we extracted the type of company and their biggest pains.

Midete gave us access to part of their database. It consisted of 5000 calls made by a company called *Agenda Pro. For context, Agenda Pro* is a company that sells and manages an operations management software to companies that deliver services that must be scheduled at a certain time of the day with anticipation, such as beauty salons or clinics. This makes taking appointments easier for clients and easier for the team to manage these appointments. Further information is provided on their web page.

The fields contained are shown in figure 4.1. The most relevant fields are shown in table 4.1. The dataset wasn't the friendliest to work with as it was not intended with the purpose of generating recommendations. It contained all of the calls made by *Agenda Pro, even* the ones that did not answer or had some problems in the call. We filtered the calls that lasted less than 30 seconds as this was considered to be too short to be relevant. We also dropped irrelevant data such as contact information, salesperson information, audio files, etc.

After the initial filtering we were left with the fields mentioned in table 4.1. and a total of 1341 calls we could extract information from. Unfortunately we were not able to gather information of which calls were follow ups from previous calls and which calls were an introduction to the product. With this in mind we could not define which calls were able to engage the clients for further information on the products, this would've given us a better understanding of which calls were successful at selling *Agenda Pro's* service. To decide which calls are labeled as successful we had to come up with our own metric based on common metrics used by the industry. This point is further explained in the Methodology section.

Methodology

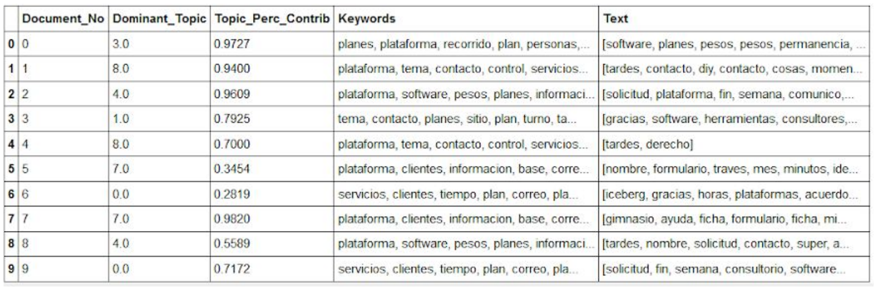
First of all we had to implement a metric or score named *successful* to determine if a call was successful or not. For this we used the parameters of talk to listen, longest monologue and longest customer story. A common industry metric is that you should talk around 40-30% of the time and listen 60-70% of it [7], other metrics commonly used are that your longest monologue should be less than a minute long and the customer longest monologue should be greater than 2 minutes long [Midete].

With this in mind we decided that a call would be successful if it achieved some of these goals. After some experimentation, we concluded that talk to listen was the most important metric and assigned a score of 3 if it was in the acceptable range (0.6 - 0.7). We assigned a score of 1 if the customers longest monologue was greater than 1 minute and a score of 1 if the salesperson's longest monologue was less than 2 minutes. Finally we implemented a binary value of 1 (successful) if the call got a score of 3 points or more and 0 (unsuccessful) if it scored less than 3. After applying this metric 378 calls were labeled as successful of a total of 1341 (28.2%). We also left the raw score as a rating value for the recommendation algorithm to use.

With the metadata that we had available, we divided the calls by the type of customer that was being contacted. To do this we attempted several approaches with different results. The first approach was to use the metadata format to our advantage since some of them contained key words like ("1- Tipo de negocio, ubicación y nº de profesionales") but few of them followed the format leaving us with only 200 calls labeled with a category (⅕ of the total dataset). We then attempted to use lemmatization and keywords per category to label them but got similar results, the main problem was that lemmatizing it deleted many keywords that could've been used. Finally, we used all the words and labeled keywords into different categories that we thought were clearly distinct leaving us with a dictionary of a business type and keywords that might reference to them. With this approach we managed to label half of the dataset. This approach may be further improved using machine learning techniques and considering the whole conversation rather than just the Metadata. Another point that might be tackled in a further study is to use a score of likelihood to assign each conversation to a business category since some words like ("paciente") could refer to many

different businesses such as psychology, medicine, veterinary, etc. *Agenda Pro* mainly dealt with 25 types of customers**.** Some of them are beauty salons (23.2%[1]), hair salons (21.9%), medical clinics (19.8%) and spa (9.9%). 595 we weren't able to identify a category.
To identify topics we first had to clean the conversations and remove special characters that might affect the algorithm like special spanish characters. Then, we filtered out stop words using the Spacy library. Finally, we lemmatized the words in order to list them as the root word (eg. "peluquería" y "peluquerías" both would point to the same word). After applying the cleaning process each transcription is now represented as a list of words spoken during the call. We use this list of words to create a corpus of all the words representing them as a bag of words, this is a tuple that contains the word id and the number of occurrences of it.

With the data cleaned we tested three different algorithms these were Latent Dirichlet Allocation (LDA), Latent Semantic Indexing (LSI) and Hierarchical Dirichlet Process (HDP). For each, we assigned 10 different topics to be extracted from all the calls, each topic is then represented by the 10 most representative words. We analyzed these topics for all the models and used a metric called coherence to compare them. With this metric HDP model turned out to be the best one but further inspection revealed that the topics didn't make much sense and the words used to describe them were not very coherent. This is why we choose LDA based on inspection. Table 5.1 shows the different topics and the keywords used to describe them.



	Document_No	Dominant_Topic	Topic_Perc_Contrib	Keywords	Text
0	0	3.0	0.9727	planes, plataforma, recorrido, plan, personas,...	[software, planes, pesos, pesos, permanencia, ...
1	1	8.0	0.9400	plataforma, tema, contacto, control, servicios...	[tardes, contacto, diy, contacto, cosas, momen...
2	2	4.0	0.9609	plataforma, software, pesos, planes, informaci...	[solicitud, plataforma, fin, semana, comunico,...
3	3	1.0	0.7925	tema, contacto, planes, sitio, plan, turno, ta...	[gracias, software, herramientas, consultores,...
4	4	8.0	0.7000	plataforma, tema, contacto, control, servicios...	[tardes, derecho]
5	5	7.0	0.3454	plataforma, clientes, informacion, base, corre...	[nombre, formulario, traves, mes, minutos, ide...
6	6	0.0	0.2819	servicios, clientes, tiempo, plan, correo, pla...	[iceberg, gracias, horas, plataformas, acuerdo...
7	7	7.0	0.9820	plataforma, clientes, informacion, base, corre...	[gimnasio, ayuda, ficha, formulario, ficha, mi...
8	8	4.0	0.5589	plataforma, software, pesos, planes, informaci...	[tardes, nombre, solicitud, contacto, super, a...
9	9	0.0	0.7172	servicios, clientes, tiempo, plan, correo, pla...	[solicitud, fin, semana, consultorio, software...

Table 5.1

We then created a baseline using the "most popular" algorithm and *success percentage*. the main difference between the two metrics is that "*success percentage*" recommends topic based on the best success-fail ratio but most popular only considers the topic with the most number of successes

Our main goal is to use recommendation algorithms to recommend topics of interest for each business category in order to increment sales. With this in mind, and given the difficult dataset to work with, our first approach is to use basic recommendation methods such as Slope-One and Funk's SVD to find out if these types of systems, regardless of their complexity, have the potential to increase the effectiveness of salesmen calls.

These algorithms are typically used to predict how well an user will rate a given item, so we had to transform the dataset to obtain users, items and ratings in order to use existing implementations of these algorithms such as python's pyreclab library. The business

[1] These percentages are based on the calls that we were able to identify with a category. 50% of the dataset was able to be identified.

category we discussed above is used as *users*, as it describes the salesman target. The topics extracted from the calls transcriptions are used as *items*, because they represent the salesman strategies to close the deal with every business category, and therefore, the selection of words that are consumed during the sales attempts, and for the *rating*, we used the *successful* metric described at the beginning of this section.

The main problem with this approach is that an user can have the same item rated multiple times, as different salesmen call different customers of the same business category and talk about the same topic, ratings are calculated based on the metrics of those particular calls and therefore, they may differ.

To solve this problem, we grouped the calls that consumed a certain topic for every business category, and assigned the rating for those particular category, topic combinations as the mean of the ratings previously given for each combination. Once we did this, we got 104 different combinations to work with.

After we obtained users, items and ratings as discussed above, we used python's *pyreclab* implementation of Slope One and SVD. First, we filtered the data in order to remove business category, topic combinations which had less than three different topics rated. After that, we divided our dataset into training and test sets, for the training set, we used the 80% of the dataset using random sampling, and the remaining 20% of the dataset for the test set. We trained and tested both models based on map@10 and ndg@10 metrics.

Parameter Analysis

Topics

Regarding the stage of topic extraction, we realised the number of topics, as well as the number of words, was very relevant and influential when trying to obtain a useful recommendation. If the word/topic selection was too broad, there were often very vague or unrelated words in the selection, or various groups of words that could be easily defined by one topic. After inspecting the extraction's results for various numbers (5, 7, 10, 15), we decided the most balanced approach for our dataset and methods was achieved with ten topics, with ten words each.

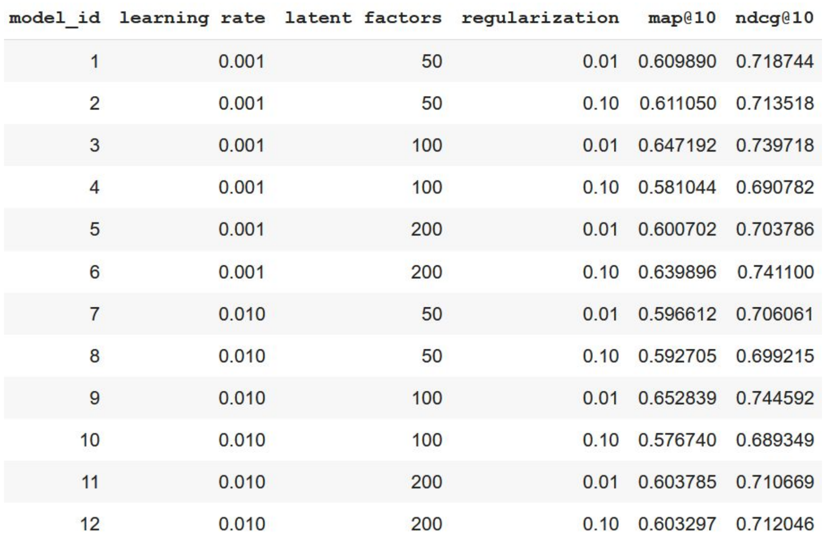
SVD

This algorithm is highly customizable, and the following results look to explore a large variety of possibilities by changing three specific parameters with each iteration, and observing the changes in the results. These parameters are learning rate, regularization and latent factors. The first describes the step size for each iteration in the process of fine-tuning the prediction. The second parameter aims to prevent overfitting, or "the production of an analysis that corresponds too closely or exactly to a particular set of data, and may therefore fail to fit additional data or predict future observations reliably". Finally, the latent factors parameter decides how specific the algorithm should be when describing the categories and topics. A higher number of latent factors implies a more thorough description of an object. [4]

Results

SVD

Table 6.1 describes all the tested combinations and their MAP and nDCG scores. In this iteration, model number 9 returned the most positive results using a learning rate of 0.01, one hundred latent factors and a regularization value of 0.01. Figures 6.1 and 6.2 give a visual representation of the consequences of changing parameters.



model_id	learning rate	latent factors	regularization	map@10	ndcg@10
1	0.001	50	0.01	0.609890	0.718744
2	0.001	50	0.10	0.611050	0.713518
3	0.001	100	0.01	0.647192	0.739718
4	0.001	100	0.10	0.581044	0.690782
5	0.001	200	0.01	0.600702	0.703786
6	0.001	200	0.10	0.639896	0.741100
7	0.010	50	0.01	0.596612	0.706061
8	0.010	50	0.10	0.592705	0.699215
9	0.010	100	0.01	0.652839	0.744592
10	0.010	100	0.10	0.576740	0.689349
11	0.010	200	0.01	0.603785	0.710669
12	0.010	200	0.10	0.603297	0.712046

Table 6.1

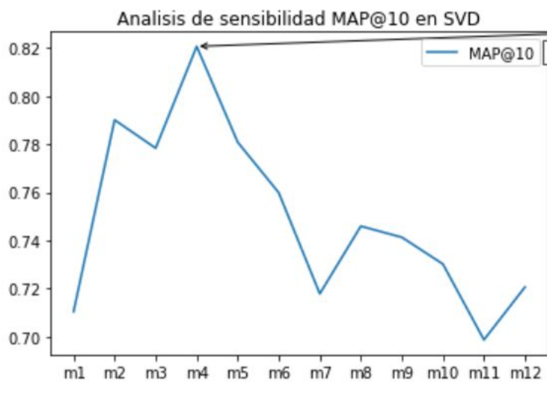


Figure 6.1

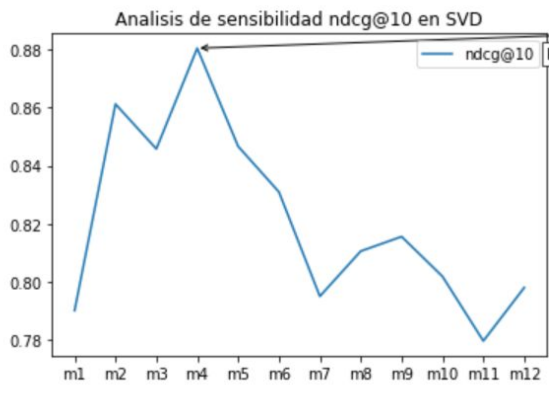


Figure 6.2

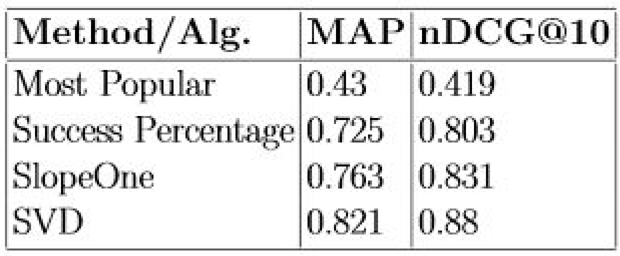
Slope One

As opposed to SVD, this algorithm does not offer the same personalization. The predictions are made by obtaining the average difference (or average slope) between all the known ratings of the items. [5]

Results Recap

It is clear that the trained algorithms offer a more precise recommendation than the Most Popular and Success Rate methods. Table 6.2 compares our results with the four methods.

Table 6.2



Method/Alg.	MAP	nDCG@10
Most Popular	0.43	0.419
Success Percentage	0.725	0.803
SlopeOne	0.763	0.831
SVD	0.821	0.88

Conclusions

Recommender Systems, regardless of their complexity, have tremendous potential for increasing the effectiveness of call centers salesmen phone calls to their customers. With basic and easy to implement algorithms such as SVD and SlopeOne, metrics like map@10 and ndcg@10 obtain decent results, nevertheless, the dataset used for this experimentation wasn't optimal, it had a lot of nonsense data and words out of place. This is mostly due to transcription software errors, which is why we are looking forward to replicating the experiments using a much larger dataset, with data that can be used without excessive pre-processing tasks as the one used in this study. Also, our results may not be explainable due to the lack of a larger amount of useful data to properly train and test our algorithms. On the other hand, with a larger and cleaner dataset, experiments with deep learning based algorithms can be made, and the results could be then compared with those obtained during this study, which can be used as a baseline.
We are very excited about the huge impact these systems may have on call-center companies. We proved that using recommender systems for this application is in fact possible. Provided a better dataset, the topics obtained would be clearly defined, making recommendations based on those topics much more explainable, which is a crucial factor for any call-center company that may want to implement this kind of systems in order to increase their sales.

References

[1] Shani, G., Rokach , L., Shapira, B., Chapnik, E., & Siboni, G. (2013). Recommending Insurance Riders.

[2] Debnath, S. (2008). Machine Learning Based Recommendation System.

[3] Industria del Call Center prevé crecer entre 4% a 6% este 2016. (2016, January)

[4] Funk, S. (2006), Try this at home. Blog post: http://sifter.org/~simon/journal/20061211.html. December 11.

[5] Lemire, D., & Maclachlan, A. (2005). Slope One Predictors for Online Rating-Based Collaborative Filtering. In SDM (Vol. 5, pp. 1-5).

[6] Padmanabhan, D., Kummamuru, K. (2007). Mining conversational text for procedures with applications in contact centers. Springer.

[7] Orlob, C. The Highest Converting Talk-to-Listen Ratio in Sales, Based on 25,537 Sales Calls (2016). Blog post: https://www.gong.io/blog/talk-to-listen-conversion-ratio/