Modeling string entries for tabular data prediction: do we need big large language models?

Léo Grinsztajn
Soda, Inria Saclay
leo.grinsztajn@inria.fr

Myung Jun Kim
Soda, Inria Saclay

Edouard Oyallon
MLIA, Sorbonne University

Gaël Varoquaux
Soda, Inria Saclay

Abstract

Tabular data are often characterized by numerical and categorical features. But these features co-exist with features made of text entries, such as names or descriptions. Here, we investigate whether language models can extract information from these text entries. Studying 19 datasets and varying training sizes, we find that using language model to encode text features improve predictions upon no encodings and character-level approaches based on substrings. Furthermore, we find that larger, more advanced language models translate to more significant improvements.

1 Context and related works

Encoding high-cardinality features Given a table with text entries, the classic One-Hot Encoding falls short when dealing with high-cardinality categories, due to the resulting dimensionality explosion. To alleviate the problem, various replacement methods have been suggested. Target Encoding is a competitive alternative that associates each category with the average value of the target variable, but it breaks when dealing with categories not seen during the training (out-of-vocabulary problem). Meanwhile, Cerda and Varoquaux [2022] have shown that character-level approaches based on substrings are competitive; these are becoming widely used as part of packages such as Catboost [Dorogush et al., 2018] or Team [2023]. These approaches, however, can only rely on the regularity in the data, as they do not incorporate any outside semantic information.

Incorporating external information Enhancing tabular data with external information, often referred to as feature enrichment, can significantly boost the prediction accuracy. If done manually, however, this process typically requires intensive labor from skilled data scientists, often involving painful joins and aggregations. To automate the process, Kanter and Veeramachaneni [2015] propose Deep Feature Synthesis, which greedily carries out joins and aggregations across tables. However, it often results in extremely high-dimensional vectors, posing challenges for effective utilization.

To mitigate this issue, subsequent research has attempted to generate useful embeddings for entities within tabular data. Cvetkov-Iliev et al. [2022] developed a method that learns embeddings from knowledge graphs. They demonstrated that such embeddings brings background information that enhances performance when incorporated into various tables. However, this approach requires a challenge of explicit matching of text entries across tables and knowledge graphs.

Language models for tabular data prediction With the widespread use of language models, several works have been proposed to enhance predictions for tabular data. Given that they are trained on huge corpora of texts, the embeddings from the language models can provide useful background

Table Representation Learning Workshop at NeurIPS 2023.

knowledge. For example, Carballo et al. [2023] observed that performance improved on one clinical dataset when using BERT-embeddings. Similarly, Cerda and Varoquaux [2022] reported competitive results when employing this approach. Moreover, the language models are robust to variations in text entries Chen et al. [2022], which solves the issue of rigorous entity matching required when incorporating external information.

Additionally, several works extend the use of language models beyond embedding entities to enhance predictions. Hollmann et al. [2023] leverages recent advancements in code generation with language models to automatically generate new features, retaining only those that boost performance. Hegselmann et al. [2023] and Dinh et al. [2022] have directly fine-tuned a language model on raw data, reporting good performance on very small datasets. Yet, these models rely on the use textual inputs and their use of background knowledge from language models on both specific entries and predictive abilities makes it challenging to disentangle their respective contributions. In this work, we show how language models can bring in background information, as opposed to string models learned on the table at hand.

2 Experimental study: text models, from simple to complex

Setup We consider 28 tabular datasets from diverse sources (A), with at least one of the column being a text entry. We want to investigate whether we can extract useful information from the text features using language models, and whether this information can be combined with other features from the same table. To this aim, we evaluate a pipeline consisting of encoding the text features using a language model, concatenating these embeddings with the corresponding tabular features, and passing these features to a classifier.

Text and numerical features processing We consider a feature to be a text feature if its cardinality is more than thirty. Other features (low cardinality categorical and numerical features) will be referred as "numerical features" for simplicity, and are encoded with the TableVectorizer from the package Team [2023], which enables automatic vectorization of a table. We use a OneHotEncoder for low cardinality variables and a MinHashEncoder [Cerda and Varoquaux, 2022] for features with a cardinality over 10. Numerical features are scaled with scikit-learn's [Pedregosa et al., 2011] StandardScaler. Regression datasets are converted to binary classification, and all dataset are balanced.

Text embedding To embed the text features, we use the SentenceTransformer package [Reimers and Gurevych, 2019], and choose models at the top of the MTEB benchmark [1] at the time of writing. The two models at top are

- BAAI's bge-large-en-v1.5 Xiao et al. [2023] [2]: a 335M parameters model pretrained on a large scale corpus, and finetuned on corpus of text pairs.
- LLMrails's ember-v1 [3]: a 335M parameters model trained on an extensive corpus of text pairs.

For these models, the embeddings are obtained by averaging the embeddings of each token (the "mean pooling" option of the package). For comparison, we also use OpenAI's embeddings [Neelakantan et al., 2022] through their API, using the model "text-embedding-ada-002", and the MinHashEncoder [Cerda and Varoquaux, 2022], a character-level approach based on substrings, available through the package Team [2023]. To reduce the dimension of text embeddings, a PCA with 30 components is used, except if using a linear model (all the embedding is used), or with the MinHashEncoder (we just encode the text with 30 components). Except specified otherwise, we use sklearn's GradientBoostingClassifier as a classifier.

2.1 Results

Text features are not always useful On the 28 datasets we consider, 11 show ROC-AUC gains of less than 1% when including the text features, compared to using only the numerical features, and 14

[1]

[2] https://huggingface.co/BAAI/bge-large-en-v1.5

[3] https://huggingface.co/llmrails/ember-v1

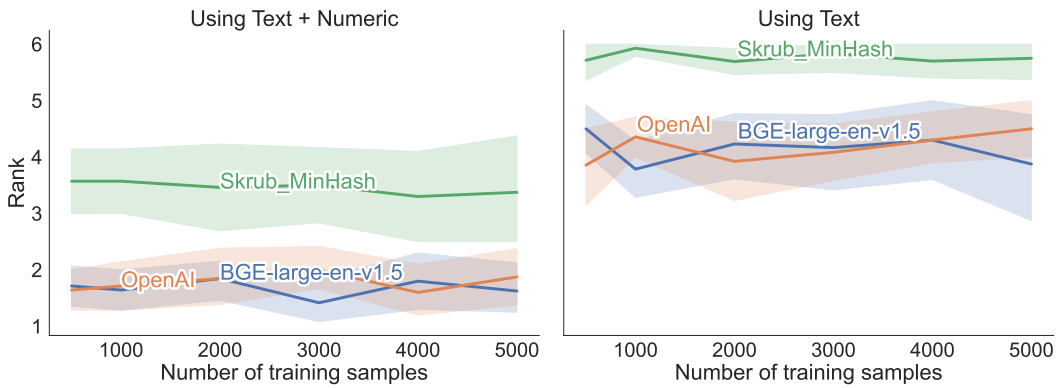
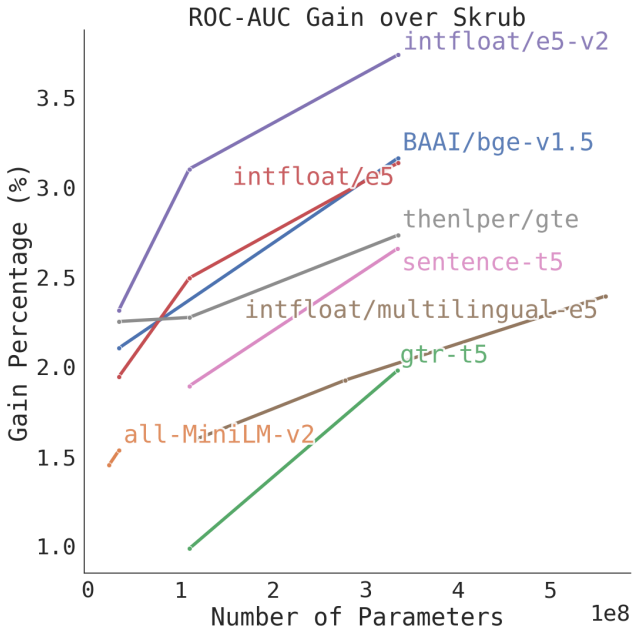


Figure 1: Comparison of different text embedding methods across varying training sizes using sklearn's GradientBoostingClassifier. The ranks are computed on 14 datasets and both settings (Text + Numeric and Text). Note that ranks are not computed across training sizes.

Figure 2: Bigger models leads to bigger gains. We vary the model used to encode text features in place of Skrub's MinHashEncoder. The gain percentage is averaged across 14 datasets, and computed on a train size of 1000, using a sklearn's GradientBoostingClassfier. Both text and numerical features are used.

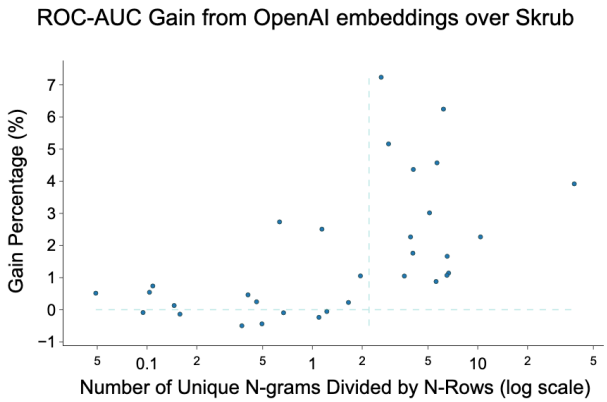


show gains of less that 3%. These gains are computed by taking the biggest gains among OpenAI embeddings, Skrub MinHashEncoder, and the 3 best models in the MTEB benchmark. In the rest of the paper, we restrict our analysis to the 14 datasets with gains above 3%.

Language models improve upon substring based methods Figure 1 shows the performance of two language model embeddings methods (OpenAI's ada-002 and BAAI's BGE-large-en-v1.5) compared to using Skrub's MinHashEncoder, a substring-based method. Across all training sizes from 500 to 5000, encoding text entries with these languages models brings clear performance over the ngram-based approach, whether using only text features, or using them in combination with numerical features.

Using bigger, better models improves performance Models in the MTEB benchmark often come in families of identical models of different sizes. In Figure 2, we take advantage of this fact to plot models from the top of the MTEB benchmark across different sizes, and observe clear gains from increasing the model size. This suggests that our pipeline will be able to benefit from both current and future advances in language models. This analysis could be extended to other model features, such as the training and finetuning data quantity.

Figure 3: For every useful text column of 14 datasets, we compute the gain from replacing Skrub's MinHashEncoder encoding of this column with OpenAI embeddings (while keeping the other columns encoded as before, see setup). The number of unique ngrams in the columns divided by the number of rows seems a good predictor of the gain. The experiment is done at a train size of 1000.



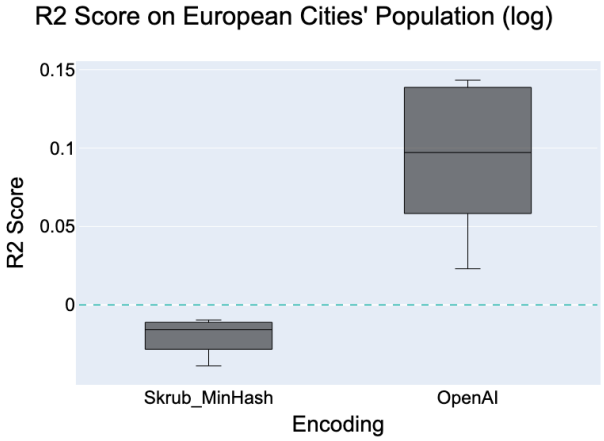


Figure 4: Comparison of the performance of Skrub's MinHashEncoder and OpenAI's embeddings for predicting the (log) population from the city and country names. Cities from a same country appear either in the train or the test set, but not in both, using sklearn's GroupKFold.

On which columns do you need language models? Investigating the distribution of gains from using language model over substring-based methods, we see that the benefits are not evenly shared. In Figure 3, we select useful columns (where prediction is more than 0.5% better when including this column with either Skrub, OpenAI, or BAAI/bge-large-en-v1.5 embeddings over dropping it) and show the gain from using language model encodings over Skrub on each column. We see approximately zero gain for around half of the columns and significant gains for the other half. A simple metric, the number of unique ngrams in the column divided by the number of rows, seems to predict well which column benefits from a language model encoding: the benefits can be seen above 2 unique ngrams per row (see Figure 3).

Language model can extract valuable knowledge from text features We think that the performance we gain from using language models to encode text entries comes from the background knowledge contained in these models [Gurnee and Tegmark, 2023]. We provide some evidence for this claim in Figure 4, where the task is to predict the population of Europeans cities (with more than 10K inhabitants) from their name, and the names of their countries. Furthermore, the train-test split is done using sklearn's GroupedKFold, such that the same country cannot appear both in the train and test set. We see that this makes it very hard for substring-based approach, as using Skrub's MinHashEncoder leads to performance akin to random chance. On the contrary, using the OpenAI embedding, we are able to retain decent performances, suggesting that we are actually using the population knowledge contained inside the embedding.

Conclusion We provide evidence that using bigger language models to represent text entries in tables capture better knowledge useful for some prediction task. This will not be true of every table: some do contains general entities, for instance in internal enterprise databases. The road ahead lies in adapting the knowledge representations to the database at hand.

References

Patricio Cerda and Gaël Varoquaux. Encoding high-cardinality string categorical variables. *IEEE Transactions on Knowledge and Data Engineering*, 34(3):1164–1176, March 2022. ISSN 1041-4347, 1558-2191, 2326-3865. doi: 10.1109/TKDE.2020.2992529.

Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. CatBoost: Gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*, 2018.

Soda Team. *Skrub: Prepping Tables for Machine Learnin*. Inria Saclay, Palaiseau, France, 2023.

James Max Kanter and Kalyan Veeramachaneni. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10, Campus des Cordeliers, Paris, France, October 2015. IEEE. ISBN 978-1-4673-8272-4. doi: 10.1109/DSAA.2015.7344858.

Alexis Cvetkov-Iliev, Alexandre Allauzen, and Gaël Varoquaux. Relational Data Embeddings for Feature Enrichment with Background Information. *Machine Learning*, 112, 2022. doi: 10.1007/s10994-022-06277-7.

Kimberly Villalobos Carballo, Liangyuan Na, Yu Ma, Léonard Boussioux, Cynthia Zeng, Luis R. Soenksen, and Dimitris Bertsimas. TabText: A Flexible and Contextual Approach to Tabular Data Representation, July 2023.

Lihu Chen, Gaël Varoquaux, and Fabian M Suchanek. Imputing out-of-vocabulary embeddings with LOVE makes language models robust with little cost. *arXiv preprint arXiv:2203.07860*, 2022.

Noah Hollmann, Samuel Müller, and Frank Hutter. Large Language Models for Automated Data Science: Introducing CAAFE for Context-Aware Automated Feature Engineering, September 2023.

Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. TabLLM: Few-shot Classification of Tabular Data with Large Language Models, March 2023.

Tuan Dinh, Yuchen Zeng, Ruisu Zhang, Ziqian Lin, Michael Gira, Shashank Rajput, Jy-yong Sohn, Dimitris Papailiopoulos, and Kangwook Lee. LIFT: Language-Interfaced Fine-Tuning for Non-language Machine Learning Tasks. *Advances in Neural Information Processing Systems*, 35: 11763–11784, December 2022.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, August 2019.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighof. C-Pack: Packaged Resources To Advance General Chinese Embedding, September 2023.

Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, Johannes Heidecke, Pranav Shyam, Boris Power, Tyna Eloundou Nekoul, Girish Sastry, Gretchen Krueger, David Schnurr, Felipe Petroski Such, Kenny Hsu, Madeleine Thompson, Tabarak Khan, Toki Sherbakov, Joanne Jang, Peter Welinder, and Lilian Weng. Text and Code Embeddings by Contrastive Pre-Training, January 2022.

Wes Gurnee and Max Tegmark. Language Models Represent Space and Time, October 2023.

Ronny Kohavi Barry Becker. Adult, 1996.

Sanjib Das, AnHai Doan, Paul Suganthan G. C., Chaitanya Gokhale, Pradap Konda, Yash Govind, and Derek Paulsen. The magellan data repository. `https://sites.google.com/site/anhaidgroup/projects/data`.

Scott Crossley, Aron Heintz, Joon Suh Choi, Jordan Batchelor, Mehrnoush Karimi, and Agnes Malatinszky. A large-scaled corpus for assessing text readability. *Behavior Research Methods*, 55 (2):491–507, 2023.

A Dataset list

We use datasets where at least one column is a text entry, and with at least 1500 rows. We right "✓" in front on datasets were text features are actually useful (see 2.1).

- **Adult** Barry Becker [1996] [4] Predict whether income exceeds $50K/yr based on census data. Also known as "Census Income" dataset.
- **Beer Profile Ratings**[5]: The dataset contains tasting profiles and consumer reviews for 3197 unique beers from 934 different breweries.
- ✓**Bikewale** Das et al. [6] Information of bikes and scooters in India. The task is to predict the degree of price of the automobiles.
- ✓**Clear Corpus** Crossley et al. [2023][7]: Generic information about the reading passage excerpts for elementary school students. The task is to predict the readability of the excerpts. The text feature is the name of the book, not the excerpt.
- ✓**Company Employees**[8]: Information on companies with over $1,000$ employees. The task is to predict the size range of the companies.
- ✓**Employee Salaries** [9]: Information on salaries for employees of the Montgomery County, MD. The task is to predict the current annual salary range of the employees.
- ✓**Employee remuneration and expenses earning over 75000** [10] Remuneration and expenses for employees earning over $75,000 per year.
- **Fifa Football Players** [11]: Football player statistics for the FIFA22 game. The task is to predict the value range of the players.
- ✓**Goodreads** Das et al. [12] Datasets containing information about books. The task is to predict the average rating for each book.
- **Japanese Anime** [13]: List of Japanese animes and their relevant information. The task is to predict the range of rating for the animes.
- ✓**Journal Influence**: Scientific journals and their descriptive features. The task is to predict the influence of a journal.
- **Michelin**[14]: List of restaurants along with additional details curated from the Michelin Restaurants guide. The task is to predict the award of the restaurants.
- **Movies**[15]: Metadata of movies released on or before July 2017. The task is to predict the range of the box-office revenues.
- **Museums**[16]: General information on the US museums. The task is to predict the range of revenues across the museums.
- ✓**Spotify**[17]: Generic information on Spotify tracks with some associated audio features. The task is to predict the popularity of the albums.

[4] https://archive.ics.uci.edu/dataset/2/adult

[5] https://www.kaggle.com/datasets/ruthgn/beer-profile-and-ratings-data-set

[6] http://pages.cs.wisc.edu/~anhai/data/784_data/bikes/csv_files/bikewale.csv

[7] https://www.commonlit.org/blog/introducing-the-clear-corpus-an-open-dataset-to-advance-research-28ff8cfea84a/

[8] https://www.kaggle.com/peopledatalabssf/free-7-million-company-dataset

[9] https://openml.org/d/42125

[10] https://opendata.vancouver.ca/explore/dataset/open-data-change-log/information/?disjunctive.datasetssort=logdaterefine.datasetids=employ remuneration-and-expenses-earning-over-75000

[11] https://www.kaggle.com/datasets/stefanoleone992/fifa-22-complete-player-dataset

[12] http://pages.cs.wisc.edu/~anhai/data/784_data/books2/csv_files/goodreads.csv

[13] https://www.kaggle.com/datasets/alancmathew/anime-dataset

[14] https://www.kaggle.com/datasets/ngshiheng/michelin-guide-restaurants-2021

[15] https://www.kaggle.com/rounakbanik/the-movies-dataset

[16] https://www.kaggle.com/datasets/markusschmitz/museums

[17] https://www.kaggle.com/datasets/maharshipandya/-spotify-tracks-dataset

- ✓**US Accidents**[18]: Information of accidents in US cities between 2016 and 2020. From this dataset, two tasks are conducted: (1) the range of accident counts for the US cities (2) the severity of the reported accidents.
- ✓**US Presidential** Cvetkov-Iliev et al. [2022]: Voting statistics in the 2020 US presidential election along with information on US counties. The task is to predict the range of voting numbers across US counties.
- **Kickstarter Projects** Projects from https://www.kickstarter.com. The task is to predict whether a project was funded.
- **Agora**[19] This is a data parse of marketplace data ripped from Agora (a dark/deep web) marketplace from the years 2014 to 2015. The task is to predict the rating.
- **Medical charges**[20]. Inpatient discharges for Medicare beneficiaries. The task is to predict the Average Total Payments.
- **NFL contracts**[21]. Contract information and draft information for NFL players from 2000-2023.
- **public**[22] Public procurement data for the European Economic Area, Switzerland, and the Macedonia. The task is to predict the award value.
- **Building Permits**[23]: Permits issued
- ✓**Ramen ratings**[24]. The dataset contain ratings and characteristics of various ramens produced from multiple countries. The task is to predict the ratings of the ramens. by the Chicago Department of Buildings since 2006. The task is to predict the Total Fee.
- **Traffic violation**[25] Traffic information from electronic violations issued in the Montgomery County, MD. The task is to predict the violation type.
- ✓**Wine reviews**
- ✓**Zomato**[26]. Information and reviews of restaurants in Bengaluru, India. The task is to predict the ratings of the restaurants.

[18] https://smoosavi.org/datasets/us_accidents
[19] https://www.kaggle.com/datasets/philipjames11/dark-net-marketplace-drug-data-agora-20142015
[20] https://www.openml.org/search?type=data&sort=runs&id=42720&status=active
[21] https://www.kaggle.com/datasets/nicholasliusontag/nfl-contract-and-draft-data
[22] https://data.europa.eu/data/datasets/ted-csv?locale=en
[23] https://www.kaggle.com/datasets/chicago/chicago-building-permits
[24] https://www.kaggle.com/datasets/residentmario/ramen-ratings
[25] https://api.openml.org/d/42132
[26] https://www.kaggle.com/datasets/himanshupoddar/zomato-bangalore-restaurants

	dataset	numeric	high_card_cat	low_card_cat	datetime
27	adult	age, fnlwgt, educational-num, ...	native-country	workclass, education, marital-...	
15	agora	Price	Vendor, Category, Item, Origin...		
26	beer_profile_and_ratings	ABV, Min IBU, Max IBU, Astring...	Name, Style, Brewery, Beer Nam...		
20	bikewale	km_driven, model_year	bike_name, color, url	city_posted, fuel_type, owner_...	
14	building_permits	PROCESSING_TIME, STREET_NUMBER...	STREET_NAME, name	PERMIT_TYPE, REVIEW_TYPE, STRE...	APPLICATION_START_DATE, ISSUE_...
8	clear_corpus	Google_WC, Sentence_Count, Par...	Author, name, Pub_Year	Categ, Lexile_Band, Location, ...	
9	company_employees	year_founded	name, domain, industry, locali...		
24	employee-remuneration-and-expe...	Year, Expenses	Name, Title	Department	
4	employee_salary	year_first_hired	department, department_name, d...	gender, assignment_category	date_first_hired
6	fifa_footballplayers_22	overall, age, height, weight, ...	name, player_positions, club_n...	preferred_foot, work_rate, bod...	
21	goodreads	PageCount, NumberofRatings	Title, Description, FirstAutho...	Format, Language	
0	journal_jcr_cls	%_of_OA_Gold	name, ISSN, eISSN, Category		
7	jp_anime	episodes, favorites, members	name, genres, studios	type, Source, Rating	
17	kickstarter	ID, goal, pledged, backers, us...	name, category	main_category, currency, count...	deadline, launched
18	medical_charge	Provider_Id, Provider_Zip_Code...	DRG_Definition, Provider_Name,...		
2	michelin		name, Location, Cuisine, Facil...		
1	movies	budget, popularity, release_da...	original_language, production_...	genres	
5	museums	Zip_Code, Locale_Code, Region_...	name, Cite, State	Museum_Type	
23	nfl_contract	draft_year, rnd, pick, g, year...	tm, player, search_key, signin...	pos	
16	public	ID_TYPE, CPV, LOTS_NUMBER, VAL...	name, CAE_ADDRESS, CAE_TOWN, C...	XSD_VERSION, CAE_TYPE, TYPE_OF...	DT_DISPATCH, DT_AWARD
25	ramen_ratings		Brand, Variety, Country	Style	
3	spotify	danceability, energy, key, lou...	name, artist	mode	
19	traffic_violations	latitude, longitude, year	seqid, description, location, ...	agency, subagency, accident, b...	date_of_stop, time_of_stop
12	us_accidents_counts		name, County, State, Airport_C...	Timezone	
11	us_accidents_severity	Distance(mi), Temperature(F), ...	County, Zipcode, Airport_Code,...	Timezone, Nearby_POI	
10	us_presidential		state, state_po, name	candidate, party	
13	wine_review	price	country, designation, province...		
22	zomato	votes	name, location, rest_type, dis...	online_order, book_table, list...	