How to predict Babies?

Thomas Berger, Pierre Bouquet, Kevin Pahud

***Abstract*—In this report, we show how to use search engine data to forecast newborns and explore different methods to obtain the best search query categories for these predictions.**

I. Introduction

Birthrate numbers are an important factor regarding the demographic activities within a country. It gives an important insight on the nation's population's "health" when defining new policies. These monthly statistics are usually released with a lag of 3 weeks in France. Receiving statistics a few weeks prior to official release could help in the development of policies, contemporaneous prediction is thus interesting. It would therefore be interesting to find google trends indicators that would allow to predict correctly these rates. However, Google Trend possesses around 1400 categories and there comes a problematic: How to define and choose the best possible categories, how many should we implement in our model and what periods should we take into account when estimating our parameter?
Our goal in this research project is to implement a few methods and design choices to obtain the best possible improvement to a base auto-regressive predictive model. We focus on two different approaches for optimal query choice. We would thus get an answer on what query is best/worst at predicting babies. It must however be noted that we do not claim any methodological advancements, the objective is to explore the benefits of using Google trend data on query choices.

II. Literature reference

Our research questions were inspired from the paper Choi *et al.* (2012), which illustrated the use of google search engine data to improve out-of-sample forecasting for economic activities using classical auto regressive models[1]. The authors focus on the implementation of seasonally adjusted search engine data to rolling window regression models to better estimate data. Their paper doesn't however show how to find the best search queries for a specific forecast nor how many must be used. There exists also a important aspect that isn't discussed within the paper: pre-processing the used datasets and the choice of the rolling window size[2].

III. Material

A. Source data

We used the statistical report on births in France released by INSEE (National institute for statistics and economical studies) for the latest month[1]. Data is imported seasonally unadjusted.

[1]https://www.insee.fr/fr/statistiques/serie/000436391#Tableau

The dataset is already pre-processed and contains all the past data. It is downloaded as a .csv file. Within the document, each number represents the amount of babies born for the given month. A clarification is made on babies born in 2020, these numbers are temporary and could be readjusted due to Covid-19 irregularities. The data is distributed on a monthly format, with statistics being released every month 3 weeks after the end of the previous month. We decided that it made sense to work with data on a log scale in the event of high fluctuations.

B. Google trends data

Google trends provide a time series index on the amount of query searches for a specific location and defined period. These series are based on categories which are broad matched (i.e. subcategories can be found in multiple categories). As our data is issued from France, we focus on extracting trends queries from France only for all categories[2]. We focus on the period 2004-2020 as it is when google trends started to record data. There exists an amount of 1400 categories that are available and a time series for each one of these is required, downloading each of them would thus be time consuming. However since there does not exist an existing API for Google trends, we used a specific library from python[3].

IV. Methodology

In terms of methodology, we had to make different choices and validations for different stakes within our study. Our proceeding method was to develop a parametric code that would run according to the different parameters (see following subsections) we applied:

A. Pre-processing

TABLE I: Source data

Period	Birth
2004-01-01	11.067248
2004-02-01	10.989048
2004-03-01	11.018383
...	...

As said previously, the source data imported required only a "classical" conversion from a .csv file to a panda format dataframe, followed by a log scale application on "Birth" values for future computation. Table I illustrates the first 3 rows (out of 202 months) of our dataframe, "Period" represents the monthly values for our birthrates. Concerning search query data, the use of an unofficial API resulted in heavy preprocessing to obtain useable data. Categories and their time series were imported as well as a dictionary (see table II) which defined a unique ID number for a specific category. This would allow the computation process to be easier for category withdrawal and selection.

[2]https://trends.google.com/trends/explore?geo=FR

[3]See https://pypi.org/project/pytrends/ for details to have an unofficial API for data extraction

TABLE II: Query dictionary

ID	Category
0	All categories
3	Arts and entertainment
5	Computer&electronics
...	...

We adapted an existing recursive function for the dictionary generation[4] as our API wasn't sufficient for easy extraction. Regarding data itself from the search query, using the unofficial API functions, a dataframe was created listing according to each month, the respective search query index for each category, see table III. Columns that were duplicates and those who were empty (not enough searches for a specific category given the chosen time period) were removed for both dataframes. We thus consequently updated both dataframes until our data was ready for analysis. However it must be noted that using the previously stated library resulted in instabilities while running our code: it is a well known problem[5].

TABLE III: Google Trends Data (GTD)

Date	0	3	5	7	...	1390	1391	1392	1397
2004-01-01	100	53	89	54	...	58	84	73	31
2004-02-01	75	53	84	55	...	85	81	92	26
...	...	...	...	...	...	...	...	...	...
2020-09-01	48	32	20	61	...	76	42	11	9
2020-10-01	52	33	21	62	...	78	43	9	10

Note: Table containing 900 categories (columns) with their IDs

B. Processing

1) Data normalisation: As a first, it was clear to think about normalising our GTD as it could potentially alter our results due to high disparities within the values, however results provided no significant changes whether or not GTD values were normalized. We still went on with this choice to optimize computational time consumption.

2) Seasonality: Seasonality is an important factor when treating econometric data[2], for instance births are high in summer and low in winter. Hence there exists an option to seasonally adjust our time series to get rid of that aspect. Since our dependant variable wasn't seasonally adjusted, it made sense not to adjust trends data.

C. Categories choice

The core idea behind the algorithm was the following: given a calculated fixed window frame k (defined according to each approach), which categories and how many of them would offer the best improvement for the upcoming month? Furthermore, should the same category be used for the whole time series (see Choi *et al.* (2012)) or should a selection be done for each month (what we define as local)? The categories selection can be done either through socio-economic studies giving indicators for birthrates[3] or through a brute force approach (i.e. test all possible categories).

In order to assess which category was optimal, we used Mean average errors (MAE) as a selection tool and defined the improvement as the relative error between the base model's MAE and the trends included model MAE.
Selection of the best category (or more) was made through two methods, correlation and direct improvement, where each of them have sub methods regarding the search conditions: deciding on which time interval the selection should be done, globally or locally.

1) Global correlation: The first method is based on the application of Pearson, Kendall and Spearman correlations between each GTD feature and the birth date on the entire date range. The correlations are then normalized in order to calculate the "correlation criterion" which is the mean between each feature's correlations, see table IV. This will thus help choose the best categories.

TABLE IV: Correlations per category on global approach

Category	Pearson	Kendall	Spearman	Mean
Ear Nose & Throat	100	94.82	93.76	96.19
Health news	83.95	100	100	94.65
...	...	...	...	...

Descending from best to worst

We then calculated the best rolling window size to use with the best feature using rolling regression for predictor estimations. Our criterion is not the best improvement but the lowest MAE Trends as the goal is too have the best model to predict the births in France.

2) Local correlation: We now decided to continue with our methods exploration, with a correlation applied on the rolling window itself this time instead of the whole time period. Protocols thus remain the same. For a given k window, we will search the best category and test for a wide range of k. Therefore the output will be a table with each column corresponding to the categories sorted by the correlation criteria during the last t^{th} month which will then be used to predict the $(t+1)^{th}$ month.

3) Global improvement function: This method implies applying the same mathematical method[6] used by Choi *et. al* (2012) (see section II). Our schism however focuses on the categories themselves, as in we test all categories available to check which one brings most improvements and try to find an optimal amount of categories to use. This is defined by a parameter n ranging from 1 to 5 (thumb rule) which would get the n best categories to check the improvements. Results for this method resulted in selecting 3 categories as an optimal choice, see table V.

TABLE V: Global improvement function results

Number of categories	1	2	3	4
Improvement [%]	3.331245	3.434077	5.864388	4.063133

[4]https://stackoverflow.com/questions/10569636/how-to-get-all-keysvalues-in-nested-dict-of-list-of-dicts-and-dicts/10569687

[5]https://github.com/GeneralMills/pytrends/issues/413

[6]https://www.statsmodels.org/stable/generated/statsmodels.regression.rolling.RollingOLS.html

4) Local improvement function: The main idea differing from the global approach is that for each month we would use the best existing category on previous month.

We would, for each iteration, store the best category (or more) of the n^{th} previous given months (defined as shift) for our prediction. As we know they're the best categories for the previous months, we just assume that they would also be good for the upcoming one. This is translated by using a fixed rolling window on the data from $t-k-\alpha$ to $t-\alpha$ to select the category (or more) with the best improvements for t, which is known, and hence use this category to predict $t+1$. k being the rolling window size and α the month shift. We are using the t^{th} value of the best category found for the y_{t-1}^{th} month. The shift comes from the following question: for how many months prior to the predicted month must we check the improvement of a certain category? Should a given category be the best one for a month or multiple? We thus included this parameter in our test which gave table VI. The best choice was 1 category and 3 months shift.

TABLE VI: Local improvements [%] according to shift vs # of categories

		Number of categories used			
		1	**2**	**3**	**4**
Shift	**1**	-8.0	-10.6	-11.5	-16.5
	2	-24.8	-19.3	-18.7	-21.5
	3	-6.0	-9.2	-11.3	-19.5
	5	-9.9	-9.3	-13.1	-16.5
	7	-14.4	-14.91	-11.1	-15.4

Both improvement function methods used k window size issued from the optimal value for the base model: As it can be seen on our table VII, selecting a rolling window of $k = 22$ gives the lowest MAE, it will thus be the parameter used for runs using the improvement function method. Note: Validation was done for k values ranging from 0 to 28.

TABLE VII: Base model errors based on rolling window

Rolling window size k	**...**	**20**	**21**	**22**	**23**
Base model MAE	...	1.9289	1.9391	1.8832	1.8864

D. Prediction model

Forecasting for each method was done the following way:

$$y_t = b_6 \cdot y_{t-6} + b_{12} \cdot y_{t-12} + b_{cat} \underbrace{best\_cat_t}_{(\gamma)} + e_t \quad (1)$$

Where γ represents the best category implemented in the model, depending on the approach. We decided to go with regressive values for 6 and 12 months prior to the prediction due to the seasonality of the birth rates.

V. Results

Now that all parameters have been validated and categories selected throughout each method, we got the following results:

A. Correlation method

As it can be seen on table VIII, using the correlation approach results in negative improvements and high errors. This stipulates that this method is not efficient for time series prediction using GTD.

TABLE VIII: Best results for prediction using correlation approach

	Best k	**Best/most_used cat**	**MAE trends**	**Improvement**
Global	30	Ear Nose & Throat	1.59018	-124.73 %
Local	36	Diabetes	1.618814	-161.0 %

Most used category is only for local approach

B. Improvement function

Regarding the improvement function method, we notice an improvement of 5.9% compared to the base model which is the only approach with a positive outcome. For the improvement function method, we got the following result:

TABLE IX: Best results for prediction using improvement fucntion

	Best k	**Best/most_used cat**	**MAE trends**	**Improvement**
Global	22	Lincoln	1.82	5.9 %
Local	22	Army	1.97	-6.0 %

Most used category is only for local approach

VI. Discussion

A few points can be made throughout this project. First of all, the best method from those of which we tested is the one most similar to Choi *et. al* (2012): using the improvement function directly but adding the fact that we test for different categories. It however did not bring high results as the best we got was a 5.9% improvement. We must note that the lowest error is with the global correlation method for a MAE of 1.59, as the rolling window is bigger. Focusing on the rolling window's size could've thus been implemented in our research project.

Another aspect that got our attention were the categories themselves. Compared to our inspiration[2], where categories are similar to the time series studied, we were not expecting categories such as *Army* or *Lincoln* to predict the best way possible. This opens up a series of question on why these categories and not *pregnancy & maternity* for example.

Furthermore, our method was based on brute force approach thus considerably increasing computation time. It could thus be annoying to test all parameters with very high computation times. However, as prediction for an upcoming month would be done once a month, computation wasn't a priority but rather different approaches.

VII. Conclusion

This research project helped us explore new ways to find Google trends data and use them to predict time series. Correlation based approach was an inefficient way to find categories whereas the improvement function method gave us positive outcomes. However, it has concluded that research should be done on optimising rolling window sizes as it influences heavily results and categories choices simultaneously. Finally, an interesting research project would be to perform socio-economic studies on why these categories are the ones that predict best birthrates.

REFERENCES

[1] Hyunyoung Choi and Hal Varian. “Predicting initial claims for unemployment benefits”. In: *Google Inc* 1 (2009), pp. 1–5.

[2] Hyunyoung Choi and Hal Varian. *Predicting the Present with Google Trends*. June 2012. URL: https://onlinelibrary.wiley.com/doi/full/10.1111/j.1475-4932.2012.00809.x.

[3] Claire Norville, Rocio Gomez, and Robert L. Brown. “Some causes of fertility rates movements”. In: *University of Waterloo, Institute*. 2003.