WS Project 1

Predicting Vaccination Uptake using Web Search Queries

Anonymous Author
Computer Science, University of Copenhagen
DIKU, KU
Copenhagen, Denmark
anonymous@ku.dk

***Abstract*—This document presents a method to predict future vaccination uptake using web-mined time-series data. The experiment carried out in this report is closely related to Hansen et al. methods for automatically predicting vaccination uptake in Denmark, by combining both clinical and web-mined data. In this document we only treat web data. My predictions on web data show a close relation with the predictions found by Hansen et al.**

Keywords—web science, vaccination uptake, google trends,web-mined data, machine learning, prediction, linear regression, web data

I. Introduction

Hansen et al. present a method for automatically predicting future vaccination uptake in Denmark, by combining clinical and web-mined data. Put simply, by considering how many people were vaccinated every month in previous years, as well as how frequently people Googled information regarding vaccinations every month in previous years, we are able to predict how many people will get vaccinated in the future on a monthly basis. This allows to reduce the reaction time of health professionals and streamline clinical resources nationwide, potentially resulting in significant financial and societal gains.

II. Methodology

A. Setup and tools used

Before we mine any kind of web data we first have to decide on what tools we need, in order to carry out the task. We are interested in data between January 2011 and December 2015. The search engine we used to download the required data is Google Trends that provides information about the frequency of query terms. Any programming language that has decent support for mining and analyzing data can be used; I went for Python 2.7 especially the *sklearn and numpy* packages which provide us with machine learning algorithms and special data structures. For downloading the reports from Google Trends we used the Selenium framework for automating our query requests and exporting to a local directory. For our experiment I decided to chose the following vaccines: *HPV* and *PCV*. The IDE of preference was PyCharm because it had some extensive configuration options that made the process of data-mining easier.

B. Selecting relevant search terms

In order to establish our query words, we need to apply a statistical method called TF-IDF (term frequency–inverse document frequency) which reflects how important a word is to a document in a collection or corpus. It is often used as a weighting factor in information retrieval and text mining. The TF-IDF value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general.

$$tfidf(t,d,D) = tf(t,d) \cdot idf(t,D)$$

$$tf(t,d) = \frac{nr.of\ times\ t\ appears\ in\ d}{total\ nr.of\ terms\ in\ d}$$

$$idf(t,D) = log_e(\frac{total\ nr.of\ documents\ D}{nr.of\ documents\ with\ term\ t})$$

Fig. 1. TF-IDF Equation

The documents in our case consists of the descriptions of the vaccines found at the following websites:

- HPV:
  - http://www.ssi.dk/Vaccination/Boernevaccination/Vaccciner%20i%20boernevaccinationsprogrammet/HPV-vaccine.aspx
  - https://www.sundhed.dk/borger/sygdomme-a-aa/kvindesygdomme/sygdomme/infektioner/hpv-vaccination/
- PCV:
  - http://www.ssi.dk/Vaccination/Boernevaccination/Vaccciner%20i%20boernevaccinationsprogrammet/Pneumokokvaccine.aspx
  - https://www.sundhed.dk/borger/sygdomme-a-aa/infektioner/sygdomme/bakteriesygdomme/invasiv-pneumokoksygdom-hos-boern/

The descriptions are written in layman terms; thus we need to apply some basic pre-processing like removing the punctuation, converting the capital letters to lowercase and removing general stop-words[1]. These words are then tokenized into individual search terms and to each term a TF-IDF algorithm is applied. These terms must satisfy a simple rule: each individual term that appears in at least two different descriptions of that vaccine can be submited for analysis.

C. Data export from Google Trends

In order to export the data from Google Trends, we have to design our very own web-scraper. After doing some searching, I discovered that Google Trends has an API feature to export data for our search terms. Unfortunately the exported CSV data isn't homogenous and not all queries have the same export format.

Some terms are exported in a week format (for ex. *2011-01-02 - 2011-01-08,0*), others in a month format (for ex. *2011-01,45*) and others might not have any data associated at all.

Also the downloaded documents have some additional information that we do not need. We need to remove the additional information and keep only the relevant data. This was done automatically importing it into a CSV Editor and removing some rows. The modified data is saved in the *sanitized* folder in the attached archive and contain only data that we need.

D. Manipulating the data

For continuing our analysis, we need to homogenize the data into the month format so we can apply our linear regression algorithms.

The terms that have been exported in *month* format are not subjected to any change because they already are in the format that we need. I decided to parse the terms that were exported in the *week* format and make an average per month. I took every entry that had the same *year-month* prefix and averaged the term frequency into one month, thus creating the same *month* format.

The query exports that had no data associated were simply removed from the analysis.

E. Ordinary Least Squares linear model

The above query frequencies are used as input features for our prediction. Even though Hansen et al. use both clinical and web data for predictions, we use only web data for our prediction. We use the provided clinical web data as ground truth. To validate our results we apply 5-fold cross-validation. Cross-validation, sometimes called rotation estimation, is a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set.

For applying our prediction algorithm I used the Ordinary Least Squares present in the *sklearn* Python machine learning package and did 5-fold cross validation on the data set. That means that our data set has been composed of a test set representing 1/5 of the entire query set, and the rest represented the training set. This is done 5 times with different parts of the data set in order to optimize parameters. The coefficient estimates for Ordinary Least Squares rely on the independence of the model terms.

The root mean squared error (*RMSE*) of a predictor measures the average of the squares of the errors or deviations, that is, the difference between the estimator and what is estimated. *RMSE* is a risk function, corresponding to the expected value of the squared error loss or quadratic loss. The difference occurs because of randomness or because the estimator doesn't account for information that could produce a more accurate estimate. We use the average *RMSE* (Fig. 2) to compare our predictor's performance. Where $\hat{p}_t$ is the prediction at time t and p_t is the real observation at time t.

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(\hat{p}_t - p_t)^2}$$

Fig. 2. RMSE equation

III. Findings

I applied the algorithm on the features that were exported by month, then by week and ultimately the combination of the two by avergaing the weeks to months. The best possible variance score is 1.0 and it can be negative (because the model can be arbitrarily worse).

A. HPV Results

For *HPV*, we get similar results to Hansen et al. if we just analyse the words that were exported by month. The average *RMSE* is very close to the findings made by Hansen et al. The features exported in weeks seem to produce worse results by averaging the weeks into months. The combination of these two sets of features produce higher a *RMSE* and a lower variance score.

I suspect that by averaging the weeks into months it might have a negative effect on the prediction. Also the two vaccine descriptions do not contain all relevant search terms, and key features might have been left out by accident when we applied the TF-IDF method in the creation of our search queries. Furthermore some week entries are missing in some weekly exports and have been replaced with 0, thus decreasing the accuracy of our predictor.

TABLE I. HPV Results

	Month			Week			Combined		
	HPV-1	HPV-2	HPV-3	HPV-1	HPV-2	HPV-3	HPV-1	HPV-2	HPV-3
RMSE:	11.524	15.165	14.416	18.773	19.092	19.200	26.343	29.632	24.768
Variance score:	0.362	-0.362	0.295	-1.044	-0.942	-0.297	-2.251	-2.881	-1.373

B. PCV Results

We had similar results for the *PCV* vaccine. Although the results are worse compared Hansen et al., this is mostly due to

[1] Stop words usually refer to the most common words in a language. (for ex. "and", "or", "but" etc.)

the size of the feature set. The query list is much smaller than the *HPV* one, because of their very short descriptions found on the the links mentioned in the Methods section. Also the factors mentioned above in the *HPV* results might impact our predictions as well.

TABLE II. PCV RESULTS

	Month			Week			Combined		
	PCV-1	PCV-2	PCV-3	PCV-1	PCV-2	PCV-3	PCV-1	PCV-2	PCV-3
RMSE:	23.587	25.444	27.605	25.100	24.536	27.326	24.313	25.737	25.696
Variance score:	-0.358	-1.036	-0.555	-0.689	-1.643	-0.917	-1.102	-2.119	-1.137

IV. CONCLUSION

We presented a method to predict future vaccination uptakes (how many people are likely to get vaccinated at some point in the near future) in Denmark by using only web-mined time-series data. As web data we used query frequencies collected from Google Trends. We have gotten very similar results to Hansel et al. in the case of the *HPV* monthly reported data, but wose results when trying to predict the *PCV* vaccination uptake.

REFERENCES

[1] N. D. Hansen, C. Lioma, and K. Molbak. Predicting vaccination uptake using web search queries. in press, 2016.