Demonstration of WP2 related software-NL

Olav ten Bosch

8 November 2016, ESSnet big data WP2, Rome

Statistics Netherlands

Technologies used

- Perl (2009), Djuggler (2010)
- Python, Scrapy (2010)
- R (2011-2015)
- Node.js (Javascript on server) (2014-)
- Google Search API (2014-)
- ElasticSearch (2016)
- Roboto (Node.js package, 2015-2016)
- Nutch: tested, not used
- Generic Framework for bulk scraping of prices
- Robot tool for detecting website changes
- Python, sklearn for ML (2016)

URL finding (1)

- Business register (BR):
  - 0.5 M enterprises with url
  - 1.0 M enterprises without url
- Sample of 1000 from enterprises *with* url
- For sample: use search on Google (API):
  - Name + ‘contact’
  - Name + Street + ‘contact’
  - Name + PostalCode + ‘contact’
  - Full address
- Save 10 results for each search => 40.000 results (max)

URL finding (2)

– Derive boolean variables for model:

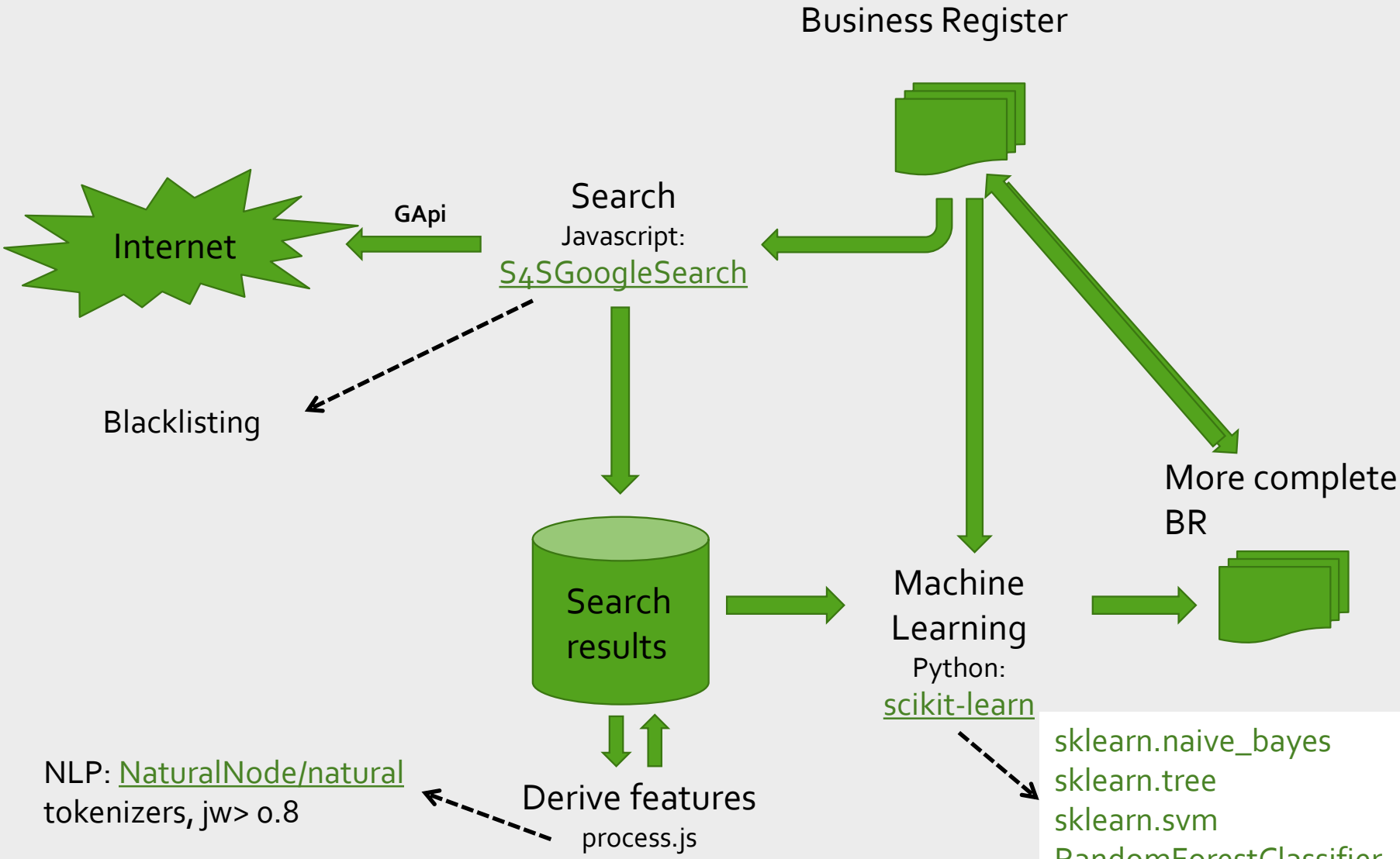
Name	Description
eqUrl	The url in the search result (foundUrl) exactly matches the url of the enterprise in our sample
eqUrlEnterpriseName	The foundUrl contains the name of the enterprise
eqTitleName	The Title contains the name of the enterprise
eqTitleAdress	The Title contains the address of the enterprise
eqTitlePostalCode	The Title contains the postal code of the enterprise
eqTitleLocality	The Title contains the locality of the enterprise
eqTitleTelephoneNr	The Title contains the telephone number of the enterprise
eqSnippetName	The Snippet contains the name of the enterprise
eqSnippetAdress	The Snippet contains the address of the enterprise
eqSnippetPostalCode	The Snippet contains the postal code of the enterprise
eqSnippetLocality	The Snippet contains the locality of the enterprise
eqSnippetTelephoneNr	The Snippet contains the telephone number of the enterprise

URL finding (3)

- Derive training set (70 %) and test set (30 %)
- Predict eqUrl from all other variables (features) using different models:
  - Naïve Bayes
  - Decision trees
  - Support Vector Machines
  - Random Forest
- Analyse results using *confusion matrix, precision*, *recall* and *f1-score*
- For about 43 enterprises we succesfully found a url according to model in the test set of 300 enterprises (~ 15 %)
- Models differ considerably in the amount of false positives and false negatives

- Apply to 1 M records => we will find 15.000 urls? (no, bias)

URL finding (4)



URL finding (4)

- Further work:
  - Refine model: experiment with other search phrases (tel.nr, Chamber of Commerce id), use other features?
  - Compare with ISTAT urlsearcher?
  - Use other search engines (or start our own)?
  - In addition to search use a focused crawler to inspect the ‘about’ page of a site (or more)
  - Derive the model from a validated set of enterprise-urls instead of a random sample from BR with url
  - Apply the model to the set of enterprises with unknown url

Search4Stats

- 2012 - 2016: Search projects for:
  - RGS (classifying business activities from product definitions)
  - Musea
  - Family businesses
  - Annual reports
  - Enterprises
- Decided to generalize and optimize search framework for statistical purposes
- Improve search modules, generic architecture

Search4Stats (S4S): generic archtecture