TweetWeather Mood Correlation

02819 - Data Mining Using Python

Woody Rousseau - Alex Pellegrini, December 9, 2013

Technical University of Denmark

Description

The project aims at finding a correlation between weather conditions and people's mood. Twitter is used as a data source from which tweets are taken and analyzed.

Application Development

The application contains four modules: `server.py`, `tweetweather.py`, `analysis.py` and `plotting.py`. They are all checked with pylint and pep8, testable aspects are done so using `py.test` and its coverage plugin. Basic profiling is performed using a custom script. The application is installable using the provided `setup.py` file, and runnable on Python 2.

1. Data Mining

Recent tweets in English are fetched using Tweepy, a Twitter API's Python wrapper. Only those including localization information are kept. Tweets and their coordinates are fetched.

For weather data, the system is interfaced with the OpenWeatherMap's API which provides real time weather condition for the localized tweets. The weather description as well as a weather icon code is fetched.

2. Data Processing

Sentiment analysis is performed by using the AFINN word value list merged with larger list. Tweet's sentiment values are scaled in the $[0.0, 1.0]$ interval. Each word is weighted by the inverse probability given by a gaussian distribution:

$$P(w) = \exp\left(-\frac{(value(w) - \mu)^2}{2\sigma^2}\right) \qquad (1)$$

This means that a tweet is also evaluated for its length implicitly. Each word contributes in value and cardinality to a category of words (`positive`, `negative` and `neutral`), whose mean gives the overall value of the tweet.

3. Data Saving

A sqlite database is used to store data, using the `pysqlite2` python module. This allows data to still be present when the application is started later.

4. Webservice

A Flask powered local webservice is used, providing several routes corresponding to html templates as well as error handlers. All previously described steps are performed on a separate daemon thread to allow the webservice to run at the same time.

5.a Table

The `/list_data` route displays all acquired data in a table sorted by descending ids. New tweets are added to the table using websockets (the `gevent-socketio` Python module) which allow the server to let the client know that new data is available.

5.b Heatmap

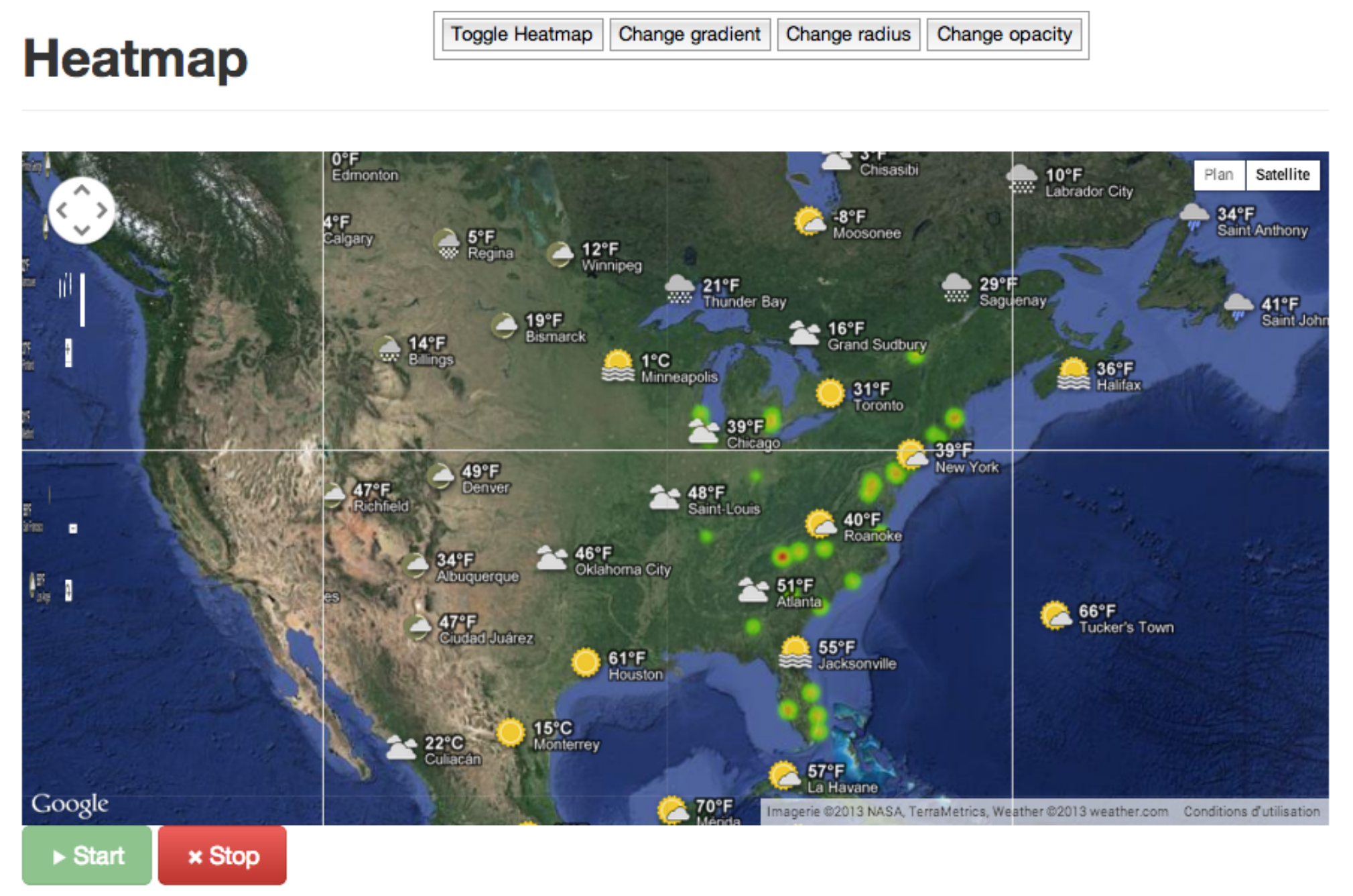
The `/map` route uses Google Map's Javascript API to display a map centered on the United States on which is added a current weather layer. When the data mining is launched, new points appear on a heatmap layer (using websockets), where the weather-sentiment correlation is used as the point weight.

5.c Scatter Plot

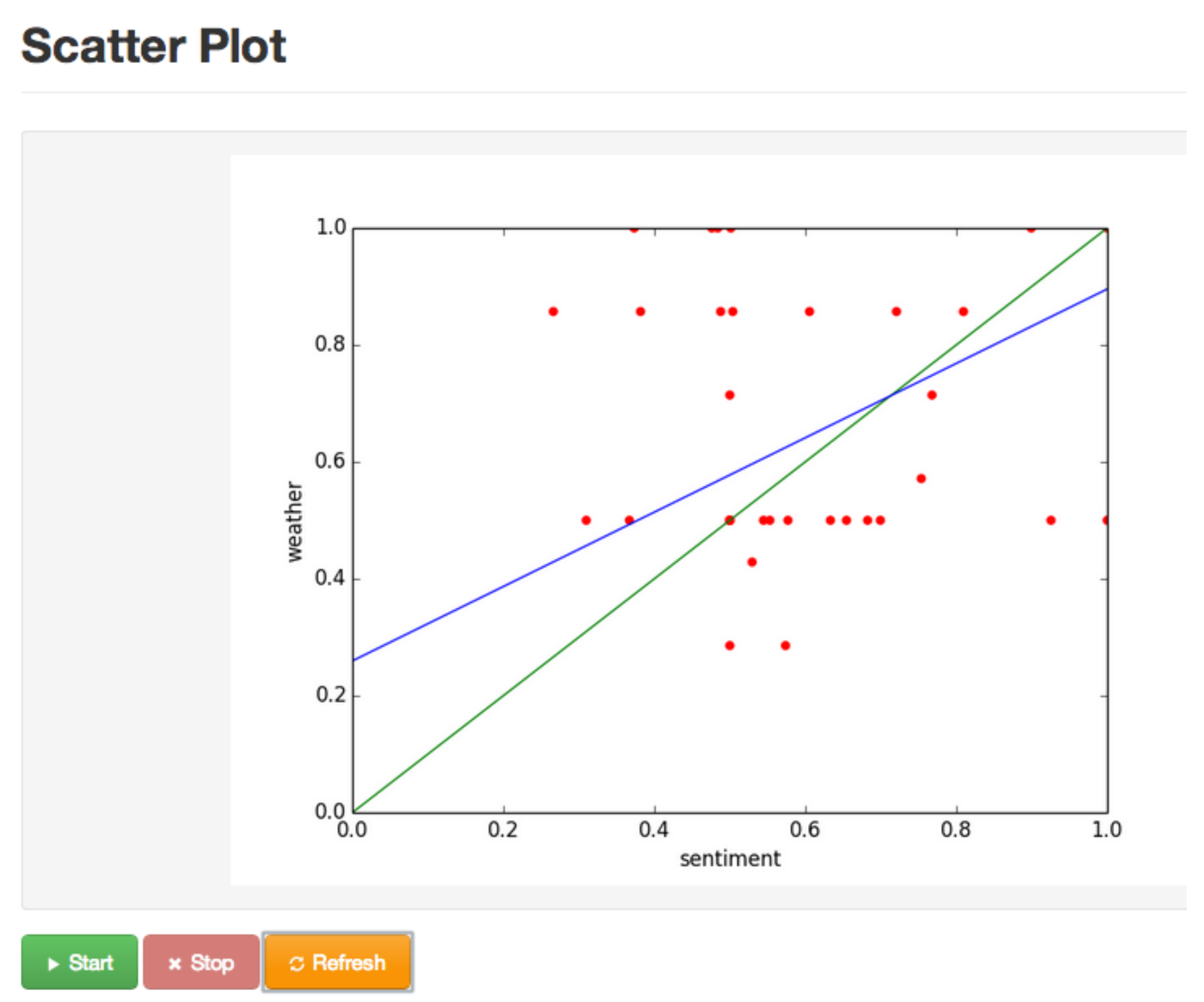
The `/plot` route uses Matplotlib and NumPy to display a scatter plot of all data points: the x-axis is the sentiment value, and the y-axis the weather value. An ideal correlation line as well as a one degree polynomial fitting is plotted.

6. Results

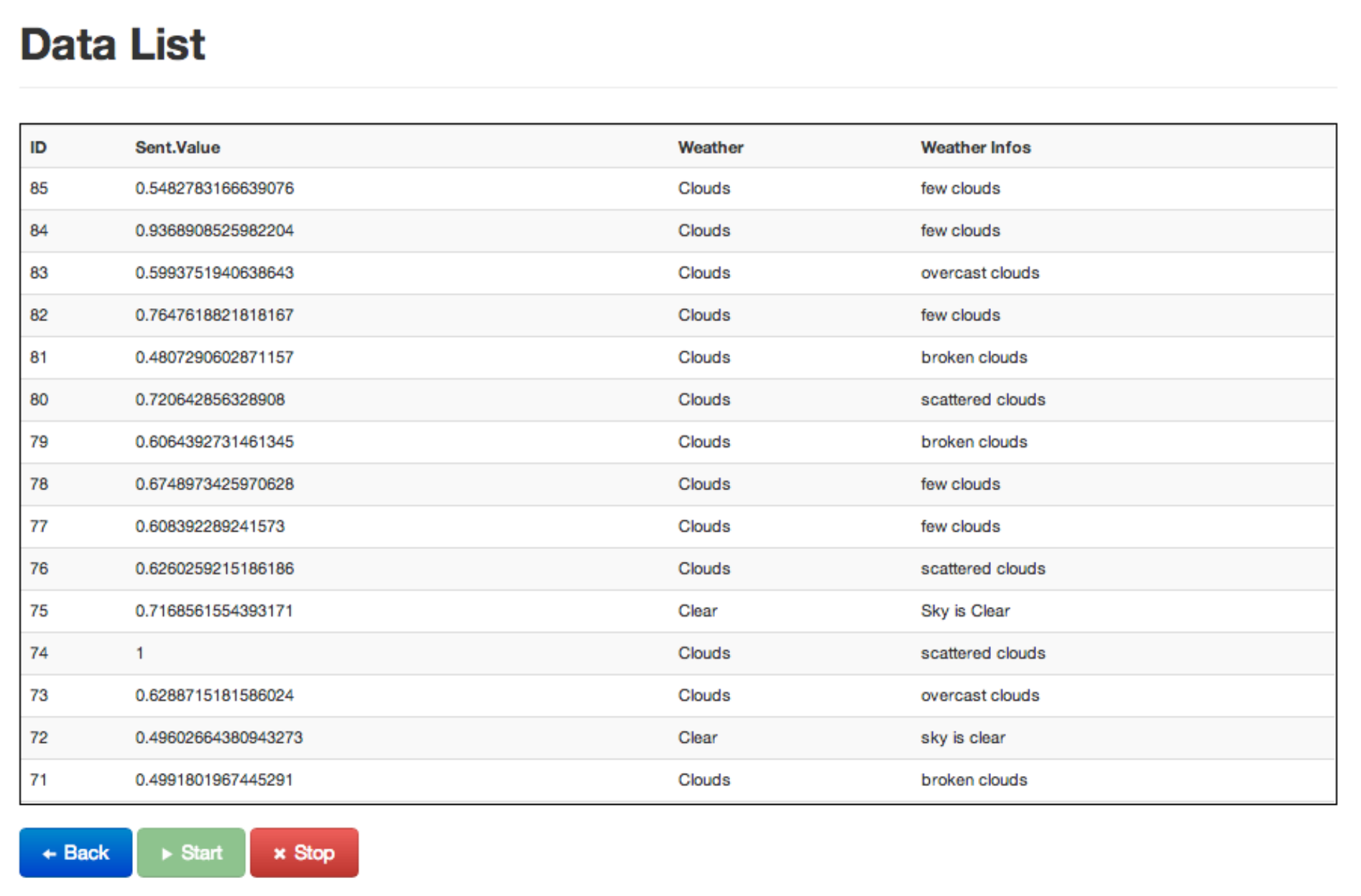
- Heatmap showing current weather conditions and new observed tweets.



- The linear correlation is far from being true when many data points are acquired.



- Dynamic list showing the last observed and analyzed data.



Data List

ID	Sent.Value	Weather	Weather Infos
85	0.5482783166639076	Clouds	few clouds
84	0.9368908525982204	Clouds	few clouds
83	0.5993751940638643	Clouds	overcast clouds
82	0.7647618821818167	Clouds	few clouds
81	0.4807290602871157	Clouds	broken clouds
80	0.720642856328908	Clouds	scattered clouds
79	0.6064392731461345	Clouds	broken clouds
78	0.6748973425970628	Clouds	few clouds
77	0.608392289241573	Clouds	few clouds
76	0.6260259215186186	Clouds	scattered clouds
75	0.7168561554393171	Clear	Sky is Clear
74	1	Clouds	scattered clouds
73	0.6288715181586024	Clouds	overcast clouds
72	0.49602664380943273	Clear	sky is clear
71	0.4991801967445291	Clouds	broken clouds

- Profiling: tweets mining is taking the longest, as localized tweets are scarce.