**How to ask Favor**

* Web to use help
* Find factors leading the success of request of online community.
* Dataset- RAOR Subreddit of Reddit.com
* Users with a single request (5728 requests)
* Average success rate- 24.6%

**Success Factors**

* Textual Factor:- Politeness, Evidentially, Reciprocity, Sentiment, Length
* Social Factors:- Status, Similarity
* Topic Modeling, Detect each word from lexicon present or absent, 15 highest scoring terms from TF-IDF weight, But topic boundaries are not very clear.
* Concise lexicon for five different narratives, simple word count to word

**Measuring the Factors**

* Temporal Factors: - Specific month, date, days, hours and community age.
* Politeness: - 19 features from computational politeness model.
* Evidentially: - Image links are detected by a regular expression, screenshot of empty bank account, a job termination letter.
* Reciprocity:- text contain related phrases like pay it forward, pay it back
* Sentiment:- Extract sentiment annotation for each sentence, further use of lexicons of positive and negative
* Status: Karma points, user account age
* Narrative:- word count, how often a given request mentions from lexicons, normalize

Develop Model

* Predict success of unseen requests
* 70% development and 30% test
* Logistic regression
* Standard likelihood ratio test to compute significance

**Result**

**Temporal factors**

Community age

First 10% request: 36.6%

Last 10% request: 16.9%

**Politeness**

Only gratitude matters

**Evidentiality**

Including an image increases chance of success

**Reciprocity**

* significantly increases the likelihood of success
* 9.9% actually do.
* express gratitude in their request return the favor 7.2%
* high status (karma in top 20%) is also positively correlated with reciprocity

**Length**

* Longer requests are significantly correlated with success.
* opportunity to provide more evidence

**Status**

* Only consider Karma point
* (We find account age to be strongly correlated with karma)
* Strongly correlated with success

**Narrative**

Correlated to the success rate

■ Student, craving : low

■ Job, money, family : high

**Interpretation**

● A short request (50 words) following the craving but no other narrative (assuming median

Karma and community age) has an estimated success probability of 9.8%.

● Using narratives that actually display more need, say the job and money narrative instead

Increases the chance to success to 19.4%,

● She puts in additional effort by writing more, say 150 words, and provides more evidence

with a picture to support her narrative. She also makes sure to display gratitude to the

community and offers to forward a pizza to someone else once she is in a better position. By

tweaking her request in a simple way she increases her chances to 56.8%

**Similarity**

* representing users by their interests in terms of the set of Subreddits in which they have

posted at least once (prior to requesting pizza),

* intersection size between the set of the giver and receiver and the Jaccard similarity

(intersection over union) of the two

* No significant correlation

**Conclusion**

* Gratitude, Reciprocity, Urgency, Status are related
* Mood/Sentiment and Similarity are not related

**Limitation**

* Cannot claim a causal relationship between the proposed factors and success that would

guarantee success.

* The set of success factors studied in this work is likely to be incomplete as well and excludes,

for instance, group behavior dynamics.

**Semi-Supervised Recognition of Sarcastic Sentences in Twitter**

**and Amazon**

**Sarcasm**

the activity of saying or writing the opposite of what you mean, or of speaking in a

way intended to make someone else feel stupid or show them that you are angry

(Macmillan English Dictionary (2007)

**Data**

One huge dataset from Amazon(66000 product reviews) and Twitter(5.9

million tweets) each.

Used the semi-supervised sarcasm identification algorithm (Tsur et al)

**Stage 1**: Perform semi supervised pattern acquisition for identifying

Sarcastic patterns a.k.a. Features

**Stage 2**: Classification stage that classifies each given sentence to a

Sarcastic category.

AMAZON:

More structured and grammatical

They come in a known context of a specific product

\*\*\*\*\* Star Rating and other meta data

Very lengthy reviews, sometimes upto, 2000 words!!.

**TWITTER:**

Lack of context and Grammar

The #sarcasm hashtag – Provides contextual reference.

Noisy data: URL, #hashtags, @usernames

Ex: “listening to Andrew Ridgley by Black Box Recorder on @Grooveshark: http://tinysong.

com/cO6i #goodmusic

“listening to Andrew Ridgley by Black Box Recorder on [USER]: [LINK] [HASHTAG].

**The Classification Algorithm**

**THE ALGORITHM IN THREE SIMPLE STEPS:**

INITIAL INPUT: a relatively small seed of labeled sentences. Labeled from 1 to 5.

Generate FEATURES from the initial input Seed.

Classify the given input sentence in question using the above knowledge

FEATURE generation: 3 steps

1. Pattern Extraction: Used an algorithm given in (Davidov and Rappoport, 2006).

CW and HFW (also includes all the punctuations and generic tags[username]

[title] etc)

Ex: “Garmin apparently does not care much about product quality or

customer support”,

“[COMPANY] CW does not CW much”, “does not CW much about CW CW

or”, “not CW much” and “about CW CW or CW CW.”

2. Pattern Selection : This is the art of wisely choosing relevant and unique patterns.

Filtered out uninformative and repetitive patterns

3. Finally, Feature Vectors, the real deal!!:

Feature vectors are the end product of the FEATURE generation process.

Patterns are converted into feature vectors.

These Feature vectors are the ones which help algorithm decide if a given sentence is

sarcastic or not.

**Data Enrichment: (AMAZON only)**

The acquired data was very limited.

Postulate : “sarcastic sentences frequently co-appear in texts with other sarcastic

sentences”

Performed an automated web search using the Yahoo! BOSS API

\*Only performed on AMAZON dataset because of its structured nature.

Steps 1,2 and 3 repeated on the new enriched data. =>result : Extended Training Set

The process of Classification!:

A test dataset consisting of to-be-labelled sentences will be passed

through the trained algorithm and then all the sentences in the test

dataset will be labelled

A KNN-like algorithm is used, which will:

Step 1 : Convert the given test set sentence into a feature vector(s)

Step 2 : Finds the nearest k neighbors of this vector compared to those

in the Extended Training Set and assigns the average rating of those k

neighbours to the given sentence.

If no match is found, a default tag of 1 (no-sarcasm) is attached.

471 positive examples and 5020 negative examples comprised the

extended training set (Amazon).

Star-sentiment baseline: Exploiting the meta data of amazon in

detection of sarcasm.

Seed training set with #sarcasm (Twitter) : 1500 tweets marked with

the #sarcasm hashtag as a positive sentence.

Seed training set (cross domain). :

for twitter the training set with #sarcasm hashtag was noisy and

inconsistent.

Therefore, created an extended set using positive examples from

Amazon dataset.

Negative examples were manually chosen from the twitter corpus.

**Evaluation & Results**

Performed in two phases:

1. A five fold cross-validation of the seed data to test its efficiency (table 2)

2. Evaluation of the efficiency of the algorithm by testing it with a new, unseen

golden standard dataset which was also being manually annotated by human

annotators and comparing the results. (table 3) TW .947\*\*