Module 12:
Opinion Mining

What is Opinion Mining?

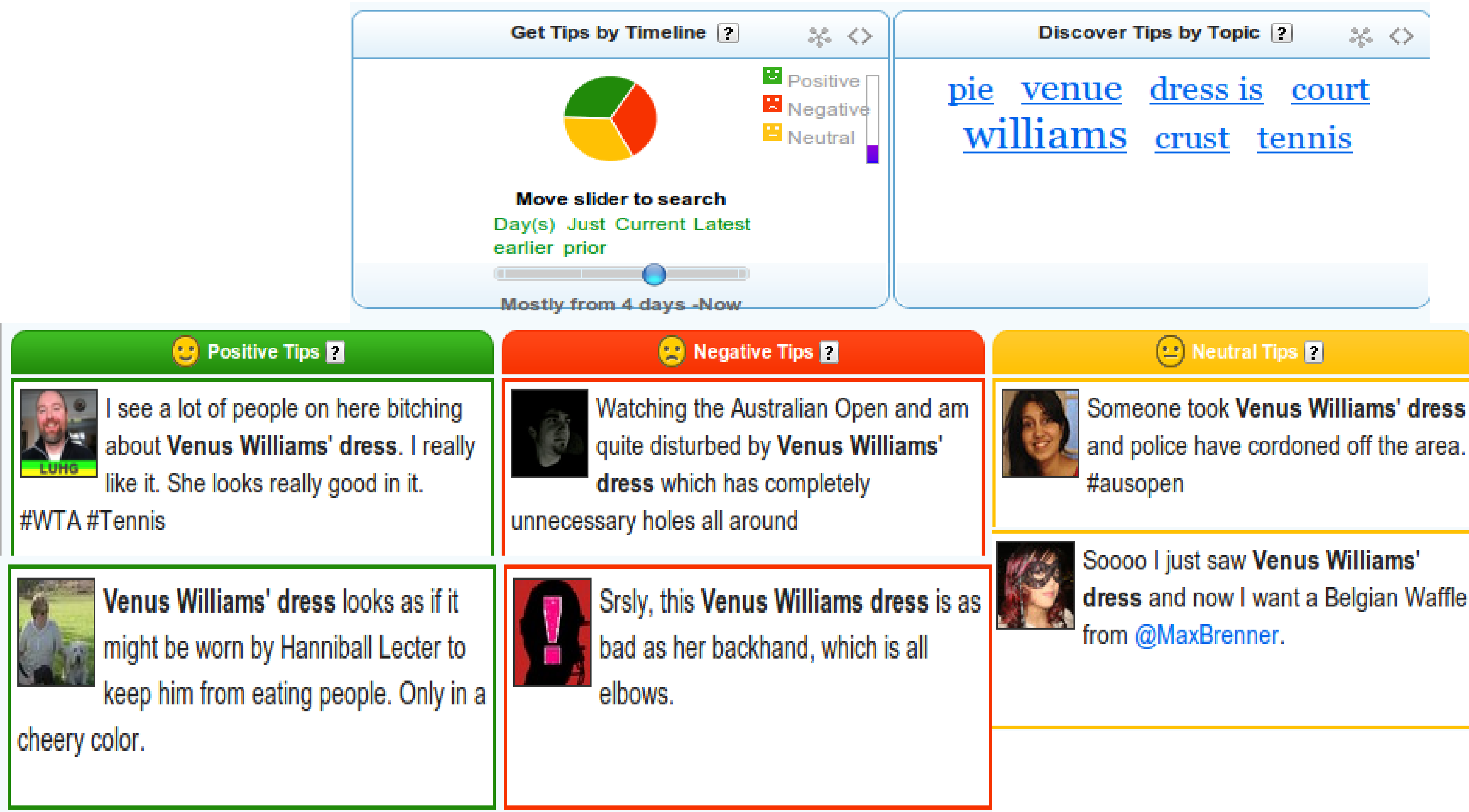
- OM is a recent discipline that studies the extraction of opinions using IR, AI and/or NLP techniques.
- More informally, it's about extracting the opinions or sentiments given in a piece of text
- Also referred to as Sentiment Analysis (though technically this is a more specific task)
- Web 2.0 nowadays provides a great medium for people to share things.
- This provides a great source of unstructured information (especially opinions) that may be useful to others (e.g. companies and their rivals, other consumers...)

It's about finding out what people think...

Opinion Mining is Big Business

- Someone who wants to buy a camera
  - Looks for comments and reviews
- Someone who just bought a camera
  - Comments on it
  - Writes about their experience
- Camera Manufacturer
  - Gets feedback from customer
  - Improve their products
  - Adjust Marketing Strategies

...and then exposing insights



What they were all talking about...

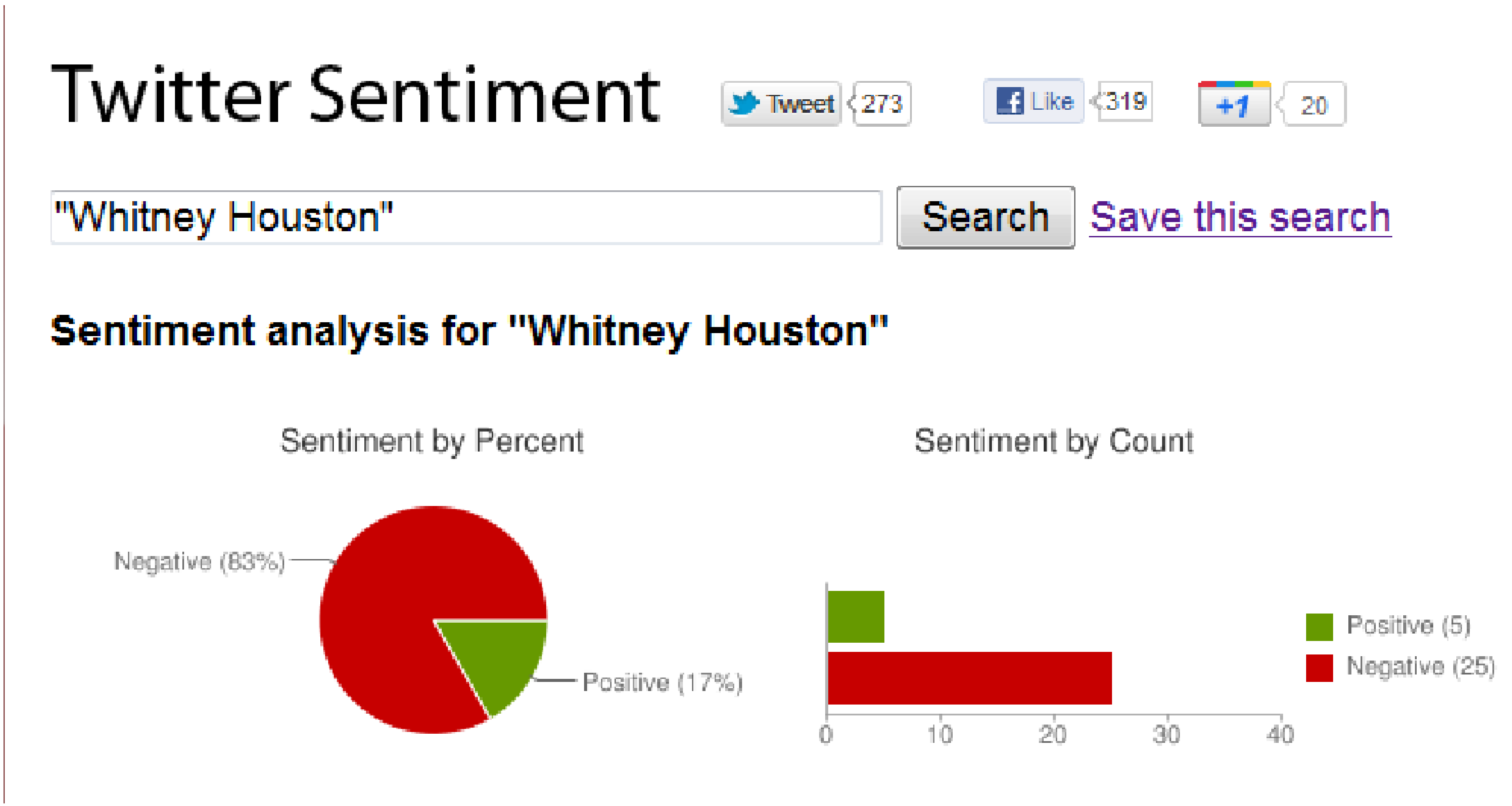
Online social media sentiment apps

- There are lots of these apps available.
- Try a search of your own on one of these::
    - Twitter sentiment http://twittersentiment.appspot.com/
    - Twends: http://twendz.waggeneredstrom.com/
    - Twittratr: http://twitrratr.com/
    - SocialMention: http://socialmention.com/
- Easy to search for opinions about famous people, brands and so on
- Hard to search for more abstract concepts, perform a non-keyword based string search

Why are these sites unsuccessful?

- They don't work well at more than a very basic level
- They mainly use dictionary lookup for positive and negative words
- They classify the tweets as positive or negative, but not with respect to the keyword you're searching for
- First, the keyword search just retrieves any tweet mentioning it, but not necessarily about it as a topic
- Second, there is no correlation between the keyword and the sentiment: the sentiment refers to the tweet as a whole
- Sometimes this is fine, but it can also go horribly wrong

Whitney Houston wasn't very popular...



Or was she?

Tweets about: "Whitney Houston"

bazzyboy25: **Whitney houston**...too soon? #CelebritiesThatLookLikeTheyStank
Posted 5 minutes ago

TeghanSimone: Radio playing **Whitney Houston**.. I swear I'm about to cry... So sad
Posted 5 minutes ago

JB3LL: hoes about to get **whitney houston**'d tonight! #TheWalkingDead
Posted 5 minutes ago

derickaadamss: "@indreamville_: Twitter I'm curious who do you think had more problems Michael Jackson or **Whitney Houston**???" <<<< **Whitney Houston**!
Posted 5 minutes ago

charlottesteer4: Listening to **Whitney Houston** loveeeee songsss <3 she's amazing <3
Posted 5 minutes ago

DionneHeraty40: @Sbarry25 The reason why **Whitney Houston** died at only 41 http://t.co/JJKRDjbj
Posted 5 minutes ago

ShortySoooFine: #musicwasbestwhen legends like James brown, Michael Jackson, **Whitney Houston** still lived.
Posted 5 minutes ago

CarlmannJohnson: Pray for Bobby Brown!!! He lost his ex-wife **Whitney Houston** and his dad Herbert Brown... Prayers up for you!!
Posted 5 minutes ago

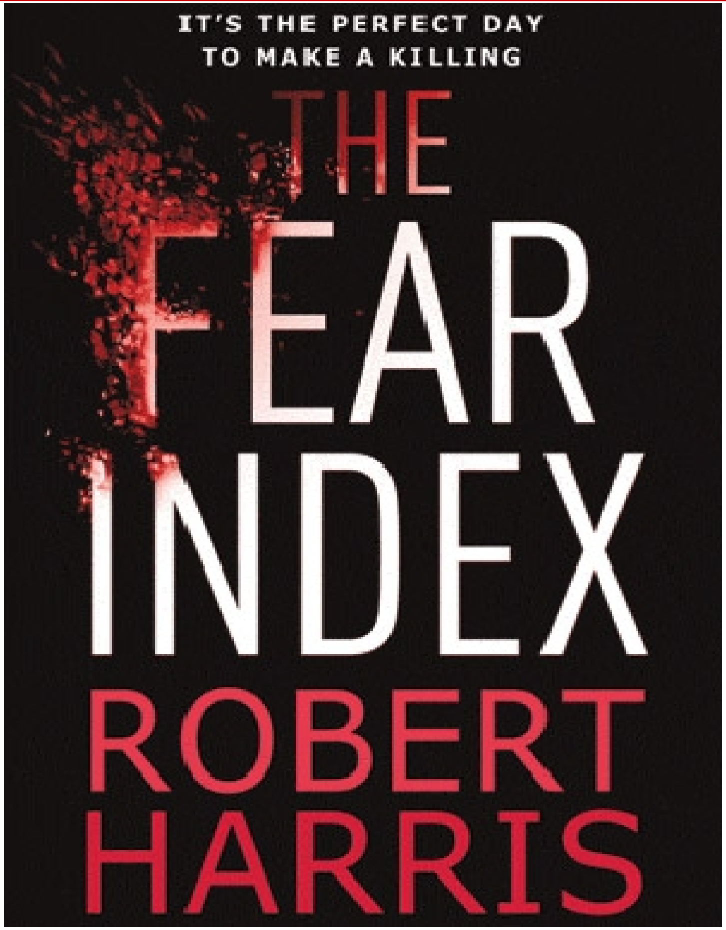
LonelySpaceman: Is it bad that I thought **Whitney Houston** was already dead?
Posted 5 minutes ago

eatmy_CHOCLATE: My aunt in there playing **Whitney Houston** making me sad
Posted 5 minutes ago

The results for this query are: Accurate Inaccur

Opinion Mining for Stock Market Prediction

- It might be only fiction, but using opinion mining for stock market prediction has been already a reality for some years
- Research shows that opinion mining outperforms event-based classification for trend prediction [Bollen2011]
- At least one investment company currently offers a product based on opinion mining



Using Twitter for Stock Market Prediction

"Hey Jon, Derek in Scunthorpe's having a bacon and egg, er, butty. Is that good for wheat futures?"

Derwent Capital Markets

- Derwent Capital Markets have launched a £25m fund that makes its investments by evaluating whether people are generally happy, sad, anxious or tired, because they believe it will predict whether the market will move up or down.
- Bollen told the Sunday Times: "We recorded the sentiment of the online community, but we couldn't prove if it was correct. So we looked at the Dow Jones to see if there was a correlation. We believed that if the markets fell, then the mood of people on Twitter would fall."
- "But we realised it was the other way round — that a drop in the mood or sentiment of the online community would precede a fall in the market."

Derwent Capital Markets

CAYMAN

" Using global sentiment analysis to trade the financial markets "

Social Media Sentiment Trading - Private Managed Accounts

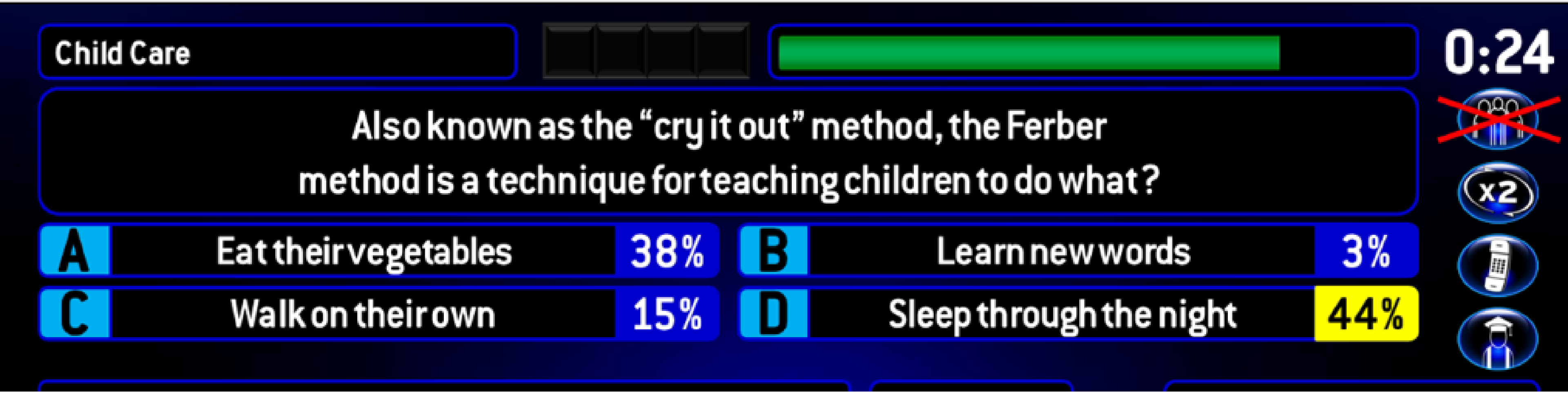
Due to massive demand we have decided to apply our social media sentiment analysis technology to managed trading accounts allowing private investors the opportunity to invest upwards of £10,000 GBP.

If you are a sophisticated investor or high net worth person and not a US resident and would like to open an account then please click the link below...

But don't believe all you read...

- It's been suggested recently that there are actually many flaws in Bollen's work, and that it's impossible to predict the stock market in this way
- If it were really possible, surely Bollen would be a millionaire by now and everyone would be using this technology?
- There's quite a lot of sloppiness in the reporting of methodology and results, so it's not clear what can really be trusted
- The advertised results are biased by selection (they picked the winners after the race and tried to show correlation)
- The accuracy claim is too general to be useful (you can't predict individual stock prices, only the general trend)
- http://sellthenews.tumblr.com/post/21067996377/noitdoesnot

Who Wants to be a Millionaire?



Ask the audience?

Or phone a friend?

Which do you think is better?

What's the capital of Spain?

A: Barcelona

B: Madrid

C: Valencia

D: Seville

What's the height of Mt Kilimanjaro?

A: 19,341 ft

B: 23,341 ft

C: 15,341 ft

D: 21,341 ft

Go for the majority or trust an expert?

- It depends what kind of question you're asking
- In Who Wants to Be a Millionaire, people tend to ask the audience fairly early on, because once the questions get hard, they can't rely on the audience getting it right

What's the height of Mt Kilimanjaro?

A: 19,341 ft
B: 23,341 ft
C: 15,341 ft
D: 21,341 ft

What's the capital of Spain?

A: Barcelona
B: Madrid
C: Valencia
D: Seville

Why bother with opinion mining?

- It depends what kind of information you want
- Don't use opinion mining tools to help you win money on quiz shows
- Recent research has shown that one knowledgeable analyst is better than gathering general public sentiment from lots of analysts and taking the majority opinion
- But only for some kinds of tasks

Whose opinion should you trust?

- Opinion mining gets difficult when the users are exposed to opinions from more than one analyst
- Intuitively, one would probably trust the opinion supported by the majority.
- But some research shows that the user is better off trusting the most credible analyst.
- Then the question becomes: who is the most credible analyst?
- Notions of trust, authority and influence are all related to opinion mining

All opinions are not equal

- Opinion Mining needs to take into account how much influence any single opinion is worth
- This could depend on a variety of factors, such as how much trust we have in a person's opinion, and even what sort of person they are
- Need to account for:
  - experts vs non-experts
  - spammers
  - frequent vs infrequent posters
  - “experts” in one area may not be expert in another
  - how frequently do other people agree?

Trust Recommenders

- Two types of trust: relationship (local) trust and reputation (global) trust.
- **Relationship trust**: if you and I both rate the same things, and our opinions on them match closely, we have high relationship trust. This can be extended to a social networking group --> web of trust.
- **Reputation trust**: if you've recommended the same thing as other people, and usually your recommendation is close to what the majority of people think, then you're considered to be more of an expert and have high reputation trust.
- We can extend relationship trust to form clusters of interests and likes/dislikes
- We can narrow reputation trust to opinions about similar topics

Related (sub)topics: general

- **Opinion extraction**: extract the piece of text which represents the opinion
  - I just bought a new camera yesterday. It was a bit expensive, but the battery life is very good.
- **Sentiment classification/orientation:** extract the polarity of the opinion (e.g. positive, negative, neutral, or classify on a numerical scale)
  - negative: expensive
  - positive: good battery life
- **Opinion summarisation**: summarise the overall opinion about something
  - price:negative, battery life: positive --> overall 7/10

Feature-opinion association

- **Feature-opinion association**: given a text with target features and opinions extracted, decide which opinions comment on which features.
  - “The battery life is good but not so keen on the picture quality”
- **Target identification**: which thing is the opinion referring to?
- **Source identification**: who is holding the opinion?
- There may be attachment and co-reference issues
  - “The camera comes with a free case but I don't like the colour much.”
  - Does this refer to the colour of the case or the camera?

Getting the target right is crucial

10 of 120 people found the following review helpful:

★★★★☆ **I'll buy this book ...**, March 15, 2010

By **T Boyer "seattleparent"** (Seattle) - See all my reviews

This review is from: The Big Short: Inside the Doomsday Machine (Hardcover)

the moment there is a 9.99 Kindle edition. I'll give it a four star rating just so I'm not drawn and quartered by the mob. (Though if you're buying a book based on average stars, without reading the reviews, well how much of a reader are you really?) I'm a big Michael Lewis fan, and I'm sorry his publisher is more interested in winning a pricing war with Amazon than with making the book available to E-book readers.

Help other customers find the most helpful reviews

Was this review helpful to you? Yes No

Report abuse | Permalink

Comments (14)

19 of 394 people found the following review helpful:

★☆☆☆☆ **Kindle Users get The Big Short !!**, March 15, 2010

By **JayRye** - See all my reviews

This review is from: The Big Short: Inside the Doomsday Machine (Hardcover)

Yes, we kindle users certainly got "The Big Short" on this title. It's really unfortunate. Kindle users take note, the Publisher is W.W. Norton and this decision to not publish a kindle version highlights that greed is not limited to the banking industry.

Help other customers find the most helpful reviews

Was this review helpful to you? Yes No

Report abuse | Permalink

Comments (14)

Opinion spamming

Suppose we run a contest where people retweet our ad repeatedly,
and the winner's whoever loses the most followers.

Spam opinion detection (fake reviews)

- Sometimes people get paid to post “spam” opinions supporting a product, organisation or even government
- An article in the New York Times discussed one such company who gave big discounts to post a 5-star review about the product on Amazon
- http://www.nytimes.com/2012/01/27/technology/for-2-a-star-a-re
- Could be either positive or negative opinions
- Generally, negative opinions are more damaging than positive ones

How to detect fake opinions?

- Review content: lexical features, content and style inconsistencies from the same user, or simlarities between different users
- Complex relationships between reviews, reviewers and products
- Publicly available information about posters (time posted, posting frequency etc)
- See anything wrong with these reviews? http://www.amazon.com/gp/pdp/profile/A3URRTIZEE8R7W

It's not just about cameras and dresses ...

- Film, theatre, books, fashion etc
  - impacts on the whole industry
  - predictions about changing society, trends etc.
- Monitoring political views
- Feedback/opinions about multimedia productions, e.g. documentaries, broadcasts etc.
- Feedback about events, e.g. conferences
- Scientific and technological monitoring, competitor surveillance etc.
- Monitoring public opinion
- Creating community memories

And it's not always as easy as it looks...

“Rubbish hotel in Madrid”

Opinion mining and social media

- Social media provides a wealth of information about a user's behaviour and interests:
  - *explicit*: John likes tennis, swimming and classical music
  - *implicit*: people who like skydiving tend to be big risk-takers
  - *associative*: people who buy Nike products also tend to buy Apple products
- While information about individuals isn't useful on its own, finding defined clusters of interests and opinions is

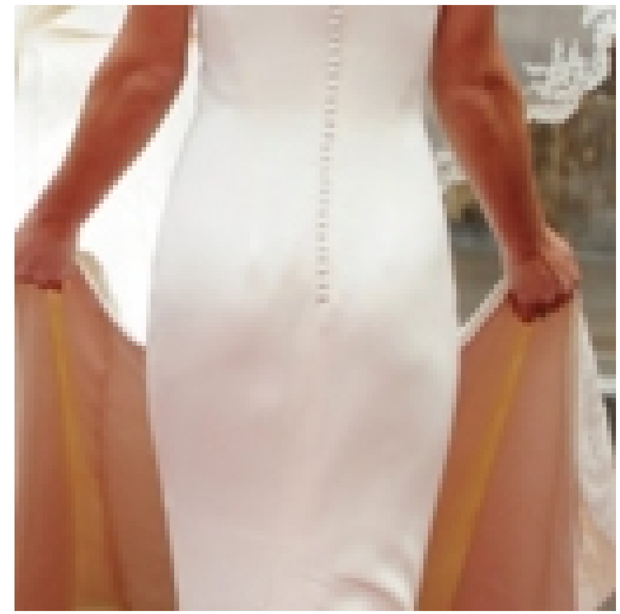
  If many people talk on social media sites about fears in airline security, life insurance companies might consider opportunities to sell a new service
- This kind of predictive analysis is all about understanding your potential audience at a much deeper level - this can lead to improved advertising techniques such as personalised ads to different groups

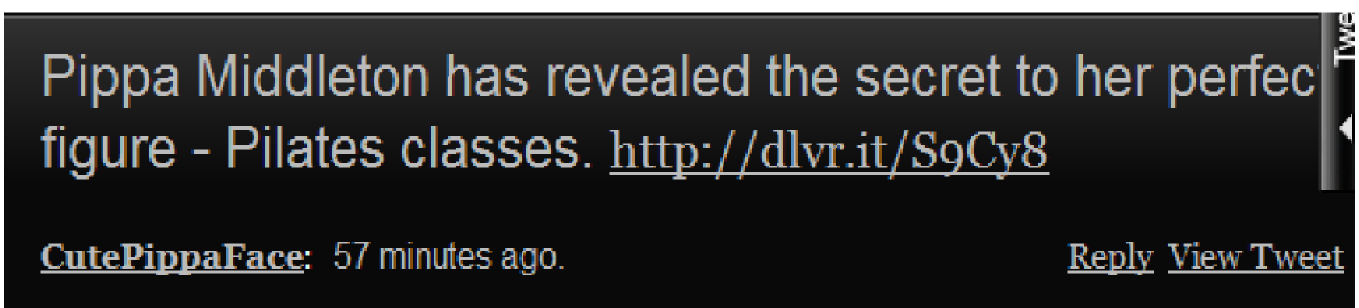
Analysing and preserving opinions

- Useful to collect, store and later retrieve public opinions about events and their changes or developments over time
- One of the difficulties lies in distinguishing what is important
- Opinion mining tools can help here
- Not only can online social networks provide a snapshot of such situations, but they can actually trigger a chain of reactions and events
- Ultimately these events might lead to societal, political or administrative changes

Pippa Middleton's assets

- One of the biggest Royal Wedding stories on Social Media sites
- Her bottom has its own twitter account, facebook page and website.
- Pilates classes have become incredibly popular since the Royal Wedding, solely as a result of all the social media





Accuracy of twitter sentiment apps

- Mine the social media sentiment apps and you'll find a huge difference of opinions about Pippa Middleton:

- TweetFeel: 25% positive, 75% negative
- Twendz: no results
- TipTop: 42% positive, 11% negative
- Twitter Sentiment: 62% positive, 38% negative

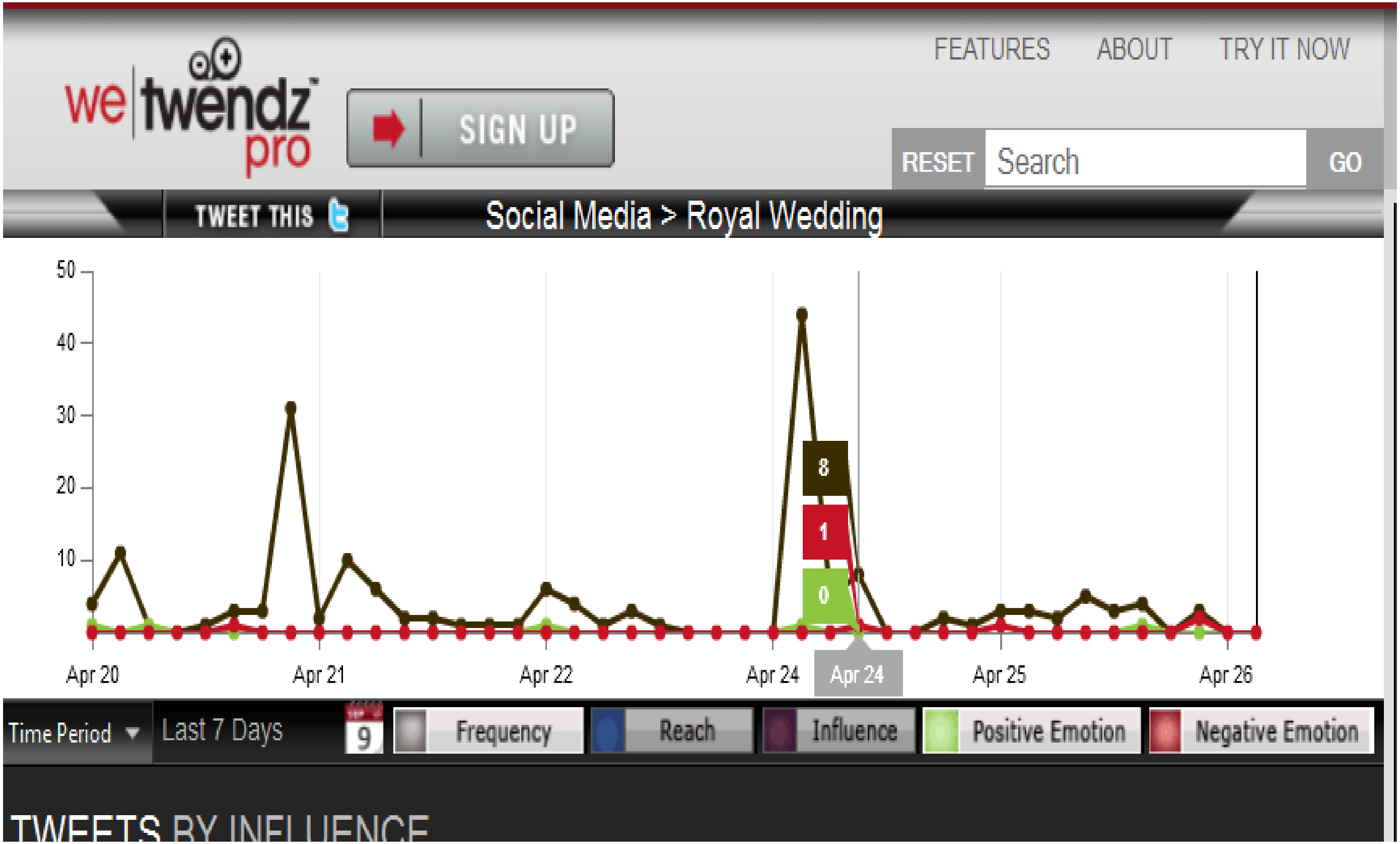
Twittrater's view of the Olympics

- A keyword search for Olympics shows exactly how existing systems fail to cut the mustard
- Lookup of sentiment words is not enough if
  - they're part of longer words
  - they're used in different contexts
  - the tweet itself isn't relevant
  - they're used in a negative or sarcastic sentence
  - they're ambiguous

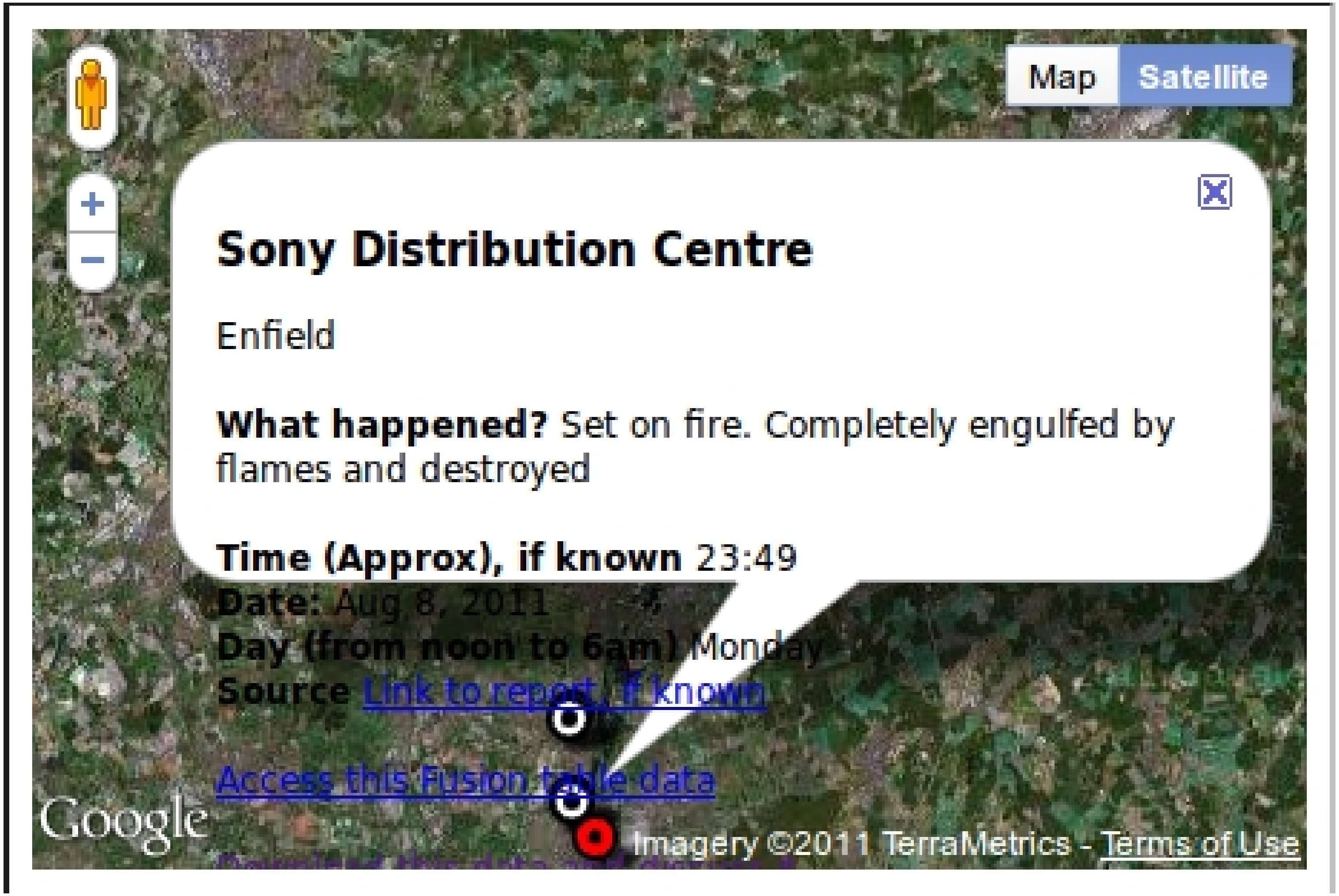
Tracking opinions over time

- Opinions can be extracted with a time stamp and/or a geo-location
- We can then analyse changes to opinions about the same entity/event over time, and other statistics
- We can also measure the impact of an entity or event on the overall sentiment about an entity or another event, over the course of time (e.g. in politics)
- Also possible to incorporate statistical (non-linguistic) techniques to investigate dynamics of opinions, e.g. find statistical correlations between interest in certain topics or entities/events and number/impact/influence of tweets etc.

Viewing opinion changes over time



Mapping dynamics from social media: UK riots demo



Rule-based Opinion Mining from Political Tweets

Processing political tweets

- Application to associate people with their political leanings, based on pre-election tweets
- First stage is to find triple <Person, Opinion, Political Party>
  - e.g. John Smith is pro_Labour
- Usually, we will only get a single sentiment per tweet
- Later, we can collect all mentions of “John Smith” that refer to the same person, and collate the information
  - For example, John may be equally in favour of several different parties, not just Labour, but hates the Conservatives above all else

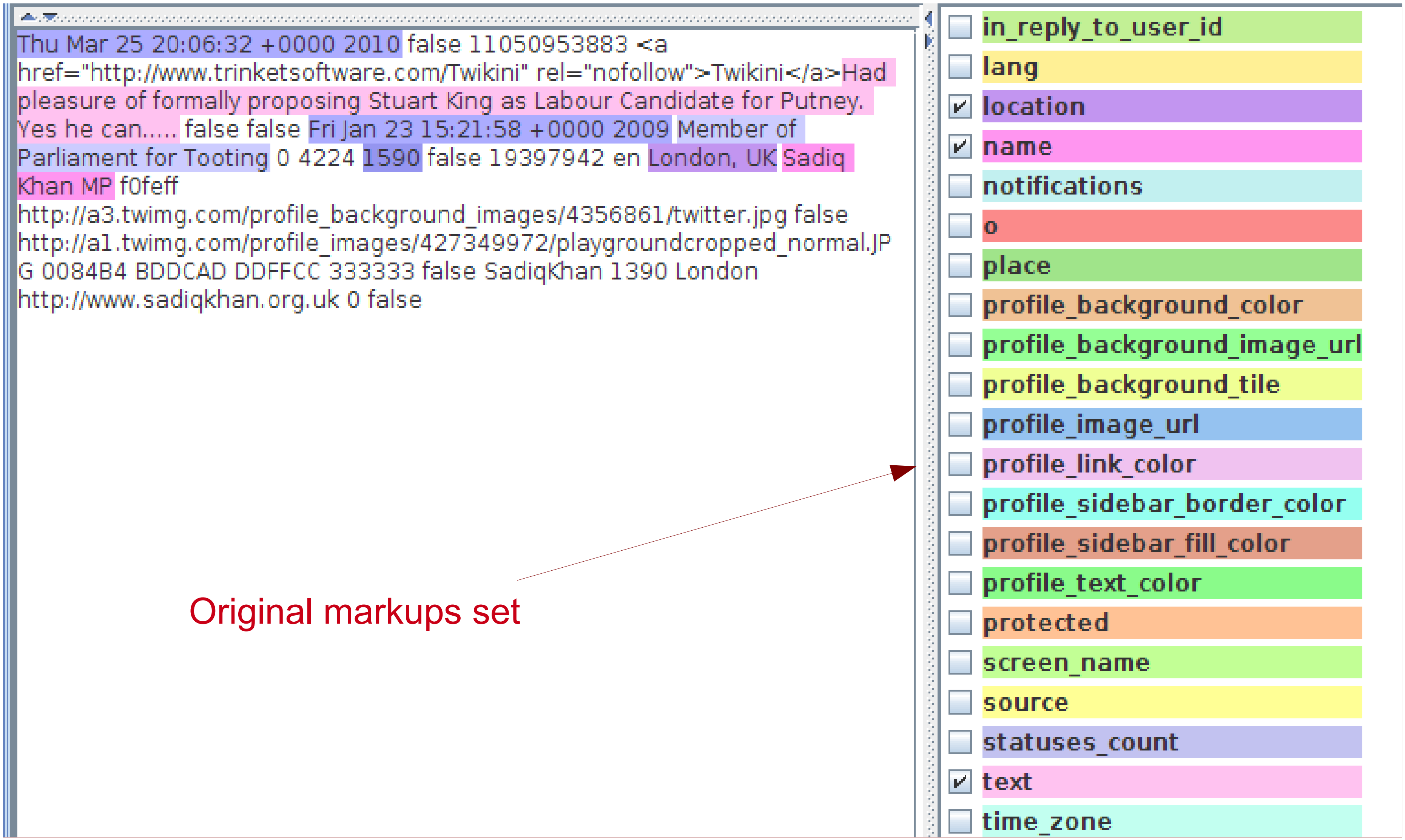
Creating a corpus

- First step is to create a corpus of tweets
- Used the Twitter Streaming API to suck up all the tweets over the pre-election period according to various criteria (e.g. use of certain hash tags, mention of various political parties etc.)
- Collected tweets in json format and then converted these to xml using JSON-:ib library
- This gives us lots of additional twitter metadata, such as the date and time of the tweet, the number of followers of the person tweeting, the location and other information about the person tweeting, and so on
- This information is useful for disambiguation and for collating the information later

Corpus Size

- Raw corpus contained around 5 million tweets
- Many were duplicates due to the way in which the tweets were collected
- Added a de-duplication step during the conversion of json to xml
- This reduced corpus size by 20% to around 4 million
- This still retains the retweets, however

Tweets with metadata



Metadata

```
Thu Mar 25 20:06:32 +0000 2010 false 11050953883 <a
href="http://www.trinketsoftware.com/Twikini" rel="nofollow">Twikini</a>Had
pleasure of formally proposing Stuart King as Labour Candidate for Putney.
Yes he can..... false false Fri Jan 23 15:21:58 +0000 2009 Member of
Parliament for Tooting 0 4224 1590 false 19397942 en London, UK Sadiq
Khan MP f0feff
http://a3.twimg.com/profile_background_images/4356861/twitter.jpg false
http://a1.twimg.com/profile_images/427349972/playgroundcropped_normal.JP
G 0084B4 BDDCAD DDFFCC 333333 false SadiqKhan 1390 London
http://www.sadiqkhan.org.uk 0 false
```

Location

Tweet

Number of friends

Name

Date

Profile info

Gazetteers

- We create an instance of a flexible gazetteer to match certain useful keywords, in various morphological forms:
  - political parties, e.g. “Conservative”, “LibDem”
  - concepts about winning election, e.g. “win”, “landslide”
  - words for politicians, e.g. “candidate”, “MP”
  - words for voting and supporting a party/ person, e.g. “vote”
  - words indicating negation, e.g. “not”, “never”
- We create another gazetteer containing affect/emotion words from WordNet.
  - these have a feature denoting part of speech (category)
  - Keeping category information may be important, so we don't want a flexible gazetteer here

Grammar rules: creating temporary annotations

- Identify questions or doubtful statements as opposed to "factual" statements in tweets.
- Initially, we just look for question marks
  - *"Wont Unite's victory be beneficial to Labour?"*
- Create temporary Affect annotations if an "affect" Lookup is found and if the category matches the POS tag on the Token (this ensures disambiguation of the different possible categories)
  - *"Just watched video about awful days of Tory rule" vs "Ah good, the entertainment is here."*
  - *"People like her should be shot." vs "People like her."*

Question grammar

```
Phase: Preprocess
Input: Token
Options: control = appelt

Rule: Question
(
 {Token.string == "?"}
):tag
-->
:tag.Question = {rule = "Question"}
```

Affect grammar

```
Phase:   Affect
Input: AffectLookup Token
Options: control = appelt

Rule: AffectAdjective
(
 {AffectLookup.category == adjective,Token.category == VBN}|
 {AffectLookup.category == adjective, Token.category == JJ}
):tag
-->
:tag.Affect = {kind = :tag.AffectLookup.kind,
      category = :tag.AffectLookup.category,  rule = "AffectAdjective"}
```

Check category of both Lookup and Token are adjectives or past participles

copy category and kind values from Lookup to new Affect annotation

Grammar rules: finding triples

- We first create temporary annotations for Person, Organization, Vote, Party, Negatives etc. based on gazetteer lookup, NEs etc.
- We then have a set of rules to combine these into pairs or triples:
  - *<Person, Vote, Party> "Tory Phip admits he voted LibDem".*
  - *<Party, Affect> "When they get a Tory government they'll be sorry."*
- We create an annotation "Sentiment" which has the following features:
  - kind = "pro_Labour", "anti_LibDem", etc.
  - opinion_holder = "John Smith", "author" etc.

Identifying the Opinion Holder

- If the opinion holder in the pattern matched is a Person or Organization, we just get the string as the value of opinion_holder
- If the opinion holder in the pattern matched is a pronoun, we first find the value of the string of the antecedent and use this as the value of opinion_holder
- Currently we only match opinion holders within the same sentence.
- If no explicit opinion holder then we use "author" as the value of opinion_holder.
- Later we can grab the details of the twitterer instead of just using "author".

Grammar rules: finding antecedents

- Find the antecedents of pronouns within a sentence so that we can refer a sentiment back to the original opinion holder or object of the opinion.
- First run the pronominal coreference PR
- Then use a JAPE rule to find pronouns linked to a Person or Organization
- We can identify these because they will have the feature “ENTITY_MENTION_TYPE” (created by the coreferencer)
- The co-referring pronouns all have also an antecedent_offset feature pointing to the proper noun antecedent
- The matching proper noun antecedent is found and its string is added as a feature on the relevant pronoun annotation

Implicit Opinion Holders

- There may not always be an explicit opinion holder
- In many cases, the author of the tweet is the opinion holder
    - *I'm also going to vote Tory. Hello new world."*
    - Here we can co-refer "I" with the person tweeting (using the metadata)
- In other cases, there is no explicit opinion holder:
    - *"Vote for Labour. Harry Potter would."*
    - However, we can infer by this instruction that the author of the tweet shares this opinion.
- In all these cases, we add the value "author" to the feature "opinion_holder"

Creating the Application

- We only want to process the actual text of the tweet, not all the other information
- To do this, we use a Segment Processing PR to run the sentiment app over just the "text" annotation in Original Markups set.
- So, we need two applications: one containing the Segment Processing PR and one containing the actual sentiment application

Runtime Parameters for the "Segment Processing PR_0001E" Segment Processing PR:

Name	Type	Required	Value
controller	CorpusController	✓	twitter app
inputASName	String		Original markups
segmentAnnotationType	String	✓	text

Hands-on 1: Analysing political tweets

GATE

- Load the ANNIE, Tools and Alignment plugins
- Load the document corpus/politwits-smiley.xml from the hands-on material and add it to a corpus
- Load the application resources/sentiment-all.gapp
- This should also load the sentiment-processing application
- Run sentiment-all.gapp on the corpus and look at the results
- Tip: Make sure you run the right application
- Click on **Sentiment** to see the overall Sentiment of the tweet
  - Sentiment.kind should be **pro-Con**
- Click on **Party** and **Vote** to see intermediate annotations

Hands-on 2: Modifying the application

- Task: modify the application to annotate the following sentence as “anti_Tory”:
  - *They all voted Tory :-(*
- Step 1: add the emoticon as a new entry in the affect gazetteer list affect_sadness.lst, with feature “category” and value “smiley”
- Step 2: add a new rule in the affect.jape grammar to create an Affect annotation from an AffectLookup with category=smiley
- Reinitialise grammar and gazetteer and rerun the application
- **Sentiment.kind** should now be **anti-Con** instead of **pro-Con**

Hands-on 3: Using ANNIC

- Create a new Lucene datastore in GATE, using the default parameters, **except** set "AnnotationSets" parameter to exclude "Key" and "Original markups".
- Create a new empty corpus, save it to the datastore, then populate it with the politwits corpus (remember to set the encoding to UTF-8 again)
- Run the application on the corpus (this may take a little while)
- Select "Lucene datastore searcher" from the datastore viewer
- Try out some patterns to see what results you get: if you find a pattern that enables you to find an opinion, try implementing it in a JAPE grammar
- Look for negative words in the tweets, and add some new gazetteer entries and/or grammar rules to deal with these.

Hands-on 3: Pattern examples

- Try thinking up your own patterns, but here are some you can try out to get you started (try changing the number of Tokens, or adding negatives):
- {Lookup.majorType == negation} ({Token})*4 {Lookup.majorType == "vote"}{Lookup.majorType == "party"}
- {Token.string == "I"} ({Token})*4 {Lookup.majorType == "vote"} {Lookup.majorType == "party"}
- {Person} ({Token})*4 {Lookup.majorType == "vote"} {Lookup.majorType == "party"}
- {Affect} ({Token})*5 {Lookup.majorType == "candidate"}
- {Vote} ({Token})*5 {Lookup.majorType == "candidate"}

Linguistic information for better analysis

- Linguistic information can give you a lot of clues about meaning
- "Good battery life" seems to indicate a positive feature.
- But conditional sentences can have subtly different meanings:
  - *I'd have bought a Nikon if I'd wanted good battery life*
  - *I'll buy a Nikon if it has good battery life*
  - *I'll buy a Nikon if I want good battery life*
  - *I'd buy a Nikon unless I wanted good battery life*
  - *I'd buy a Nikon even if it doesn't have good battery life.*

Conditional Types

0. If a camera has 20 hours of battery life, you can take many pictures.
    - statement of fact or certainty
1. If someone makes a camera with 20 hours of battery life, I'll buy it
    - potential conditional
    - *long battery life is my top priority*
2. If someone made a camera with 20 hours of battery life, I'd buy it
    - less probably conditional. Indicates preference
    - *as (1), but I think it's unrealistic so I'll settle for something else*
3. If someone had made a camera with 20 hours of battery life, I'd have bought it
    - Impossible past events
    - *as (1), but they don't make one, so I bought something else*

More examples

2. If I wanted a camera with 20 hours of battery life, I would buy a Nikon
   - *battery life is not my priority, so I'll probably buy something else*
3. If I had wanted a camera with 20 hours of battery life, I'd have bought a Nikon
   - *battery life is not my priority and so I bought something other than Nikon*

Linguistic analysis of conditional types

Type 0: If + simple present --> simple present

- *If it has good battery life, you can take lots of pictures*

Type 1: If + simple present --> simple future

- *If it has good battery life, I will buy it*

Type 2: If + past --> would + infinitive

- *If it had good battery life, I would buy it*

Type 3: If + past perfect --> present perfect

- *If it had had good battery life, I would have bought it*

Simple conditional application in GATE

- Gazetteer list gives us words associated with conditionals
  - if, assuming, even if, as long as, on condition that... (positive)
  - unless (negative)
- Verb chunker segments the VPs and also gives
  - the tense of the verb
  - active or passive
  - positive or negative
- Grammar rules combine items from gazetteer with verb information to create rules for sentences

Sample grammar for type 0 conditional

Pattern: If + simple present, simple present

```
Input: Split VG ConditionalIndicator

Rule: Conditional0
(
 {ConditionalIndicator}
 {VG.tense == SimPre}
 {VG.tense == SimPre}
 {Split}
):tag
-->
:tag.Conditional = {type = "0"}
```

{Lookup.majorType == conditional}

Verb phrases with verb in the simple present tense

Don't let the pattern span a sentence boundary

Tag the whole sentence as a conditional of type 0

Why do we do the Lookup in a separate phase?

- Why do we first find the Conditional Lookups and annotate them separately? Why not just use the Lookup annotation within the rule?
- The clue is in the Input headers
- If we use a Lookup annotation within the rule, we need to add "Lookup" to the Input headers
- What effect might this have on the rule?
- Remember that we only want to state explicitly in the rule the things we care about.
- We don't care (at this stage) which nouns occur in the sentence so we want to leave as much as possible unspecified.

Hands-on 3: conditionals

- Remove all loaded applications and documents from GATE
- Load the application resources/conditionals.gapp from the hands-on materials
- Load the document corpus/conditional-sentences.txt, add to a corpus and run the application on it
- Check the results
- Have a look at the grammar conditional-polarity.jape and see if you can work out how the negation part works

Negation: adding the polarity feature

- The sentence is divided into its two verb phrases: firstPol and secondPol
- For each phase, if the value of the neg feature is “yes”, then “neg” is stored as the new value
- If the value of the neg feature is “no”, then “pos” is stored as the new value
- A new feature called “polarity” is added to the final annotation that covers the whole sentence
- The values of the two neg features (one for each VP) are added consecutively as the values of polarity, e.g. “neg” + “pos”

Application: the Arcomem system

- Developed a series of initial applications for opinion mining from social media using GATE
- Based on the previous application for identifying political opinions from tweets
- Extended to more generic analysis about any kind of entity or event, in 2 domains
  - Greek financial crisis
  - Rock am Ring (German rock festival)
- Uses a variety of social media including twitter, facebook and forum posts
- Based on entity and event extraction, and a rule-based approach

Why Rule-based?

- Although ML applications are typically used for Opinion Mining, this task involves documents from many different text types, genres, languages and domains
- This is problematic for ML because it requires many applications trained on the different datasets, and methods to deal with acquisition of training material
- Aim of using a rule-based system is that the bulk of it can be used across different kinds of texts, with only the pre-processing and some sentiment dictionaries which are domain and language-specific

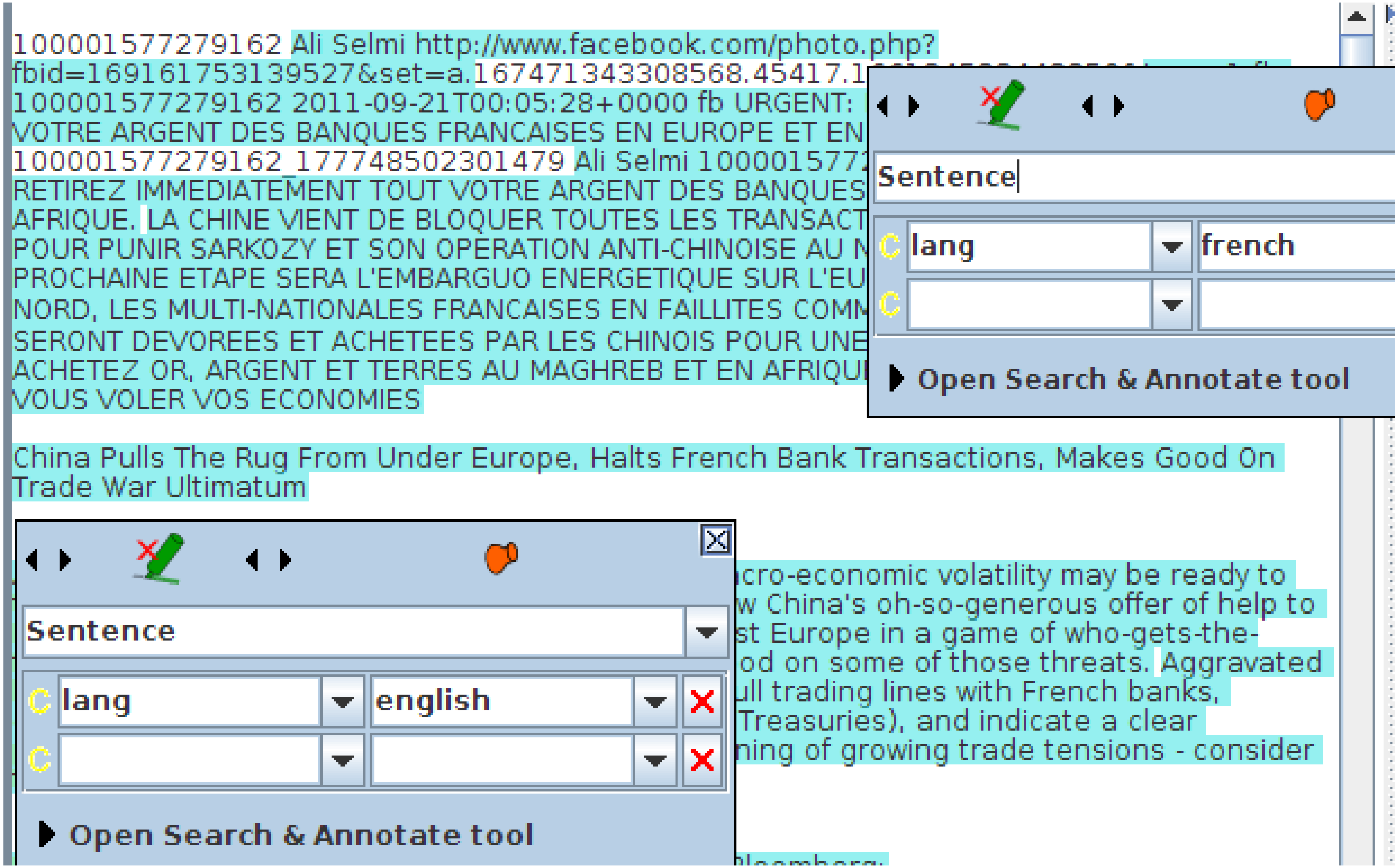
Application Stages

- Structural pre-processing, specific to social media types
- Linguistic pre-processing (including language detection), NE, term and event recognition
- Additional targeted gazetteer lookup
- JAPE grammars
- RDF-XML generation
- Aggregation of opinions
- Dynamics

Linguistic pre-processing

- Language identification (per sentence) using TextCat
- Standard tokenisation, POS tagging etc using GATE
- Modified versions of ANNIE and TermRaider for NE and term recognition
- Event recognition using specially developed GATE application (e.g. band performance, economic crisis, industrial strike)

Language ID with TextCat



Basic approach for opinion finding

- Find sentiment-containing words in a linguistic relation with entities/events (opinion-target matching)
- Use a number of linguistic sub-components to deal with issues such as negatives, irony, swear words etc.
- Starting from basic sentiment lookup, we then adjust the scores and polarity of the opinions via these components

Sentiment finding components

- **Flexible Gazetteer Lookup**: matches lists of affect/emotion words against the text, in any morphological variant
- **Gazetteer Lookup:** matches lists of affect/emotion words against the text only in non-variant forms, i.e. exact string match (mainly the case for specific phrases, swear words, emoticons etc.)
- **Sentiment Grammars**: set of hand-crafted JAPE rules which annotate sentiments and link them with the relevant targets and opinion holders
- **RDF Generation**: create the relevant RDF-XML for the annotations according to the data model (so they can be used by other components)

Opinion scoring

- Sentiment gazetteers (developed from sentiment words in WordNet) have a starting “strength” score
- These get modified by context words, e.g. adverbs, swear words, negatives and so on

Challenges imposed by social media

- **Language**: specific pre-processing for Twitter. use shallow analysis techniques with back-off strategies; incorporate specific subcomponents for swear words, sarcasm etc.
- **Relevance**: topics and comments can rapidly diverge. Solutions involve training a classifier or using clustering techniques
- **Target identification**: use an entity-centric approach
- **Contextual information**: use metadata for further information, also aggregation of data can be useful

Opinion Finding in Different Tasks

Corpus	Sentiment detection	Polarity detection	Target assignment
Political Tweets	78%	79%	97.9%
Financial Crisis Facebook	55%	81.8%	32.7%
Financial Crisis Tweets	90%	93.8%	66.7%

In general, the low numbers are due to harder tasks!

Machine Learning for Sentiment Analysis

- ML is an effective way to classify opinionated texts
- We want to train a classifier to categorize free text according to the training data.
- Good examples are consumers' reviews of films, products, and suppliers.
- Sites like www.pricegrabber.co.uk show reviews and an overall rating for companies: these make good training and testing data
- We train the ML system on a set of reviews so it can learn good and bad reviews, and then test it on a new set of reviews to see how well it distinguishes between them
- We give an example of a real application we developed for a previous project, and some related hands-on for you to try

Examples of consumer reviews

Merchant Info | Merchant Ratings | Uncategorized Products

Sort Reviews by: Date Rating

Write a Review »

Date Reviewed: 16/04/08

poet2000 **Member Since:** 16/04/08 **View Member's:** Reviews	30 days and still waiting	**Overall Rating**

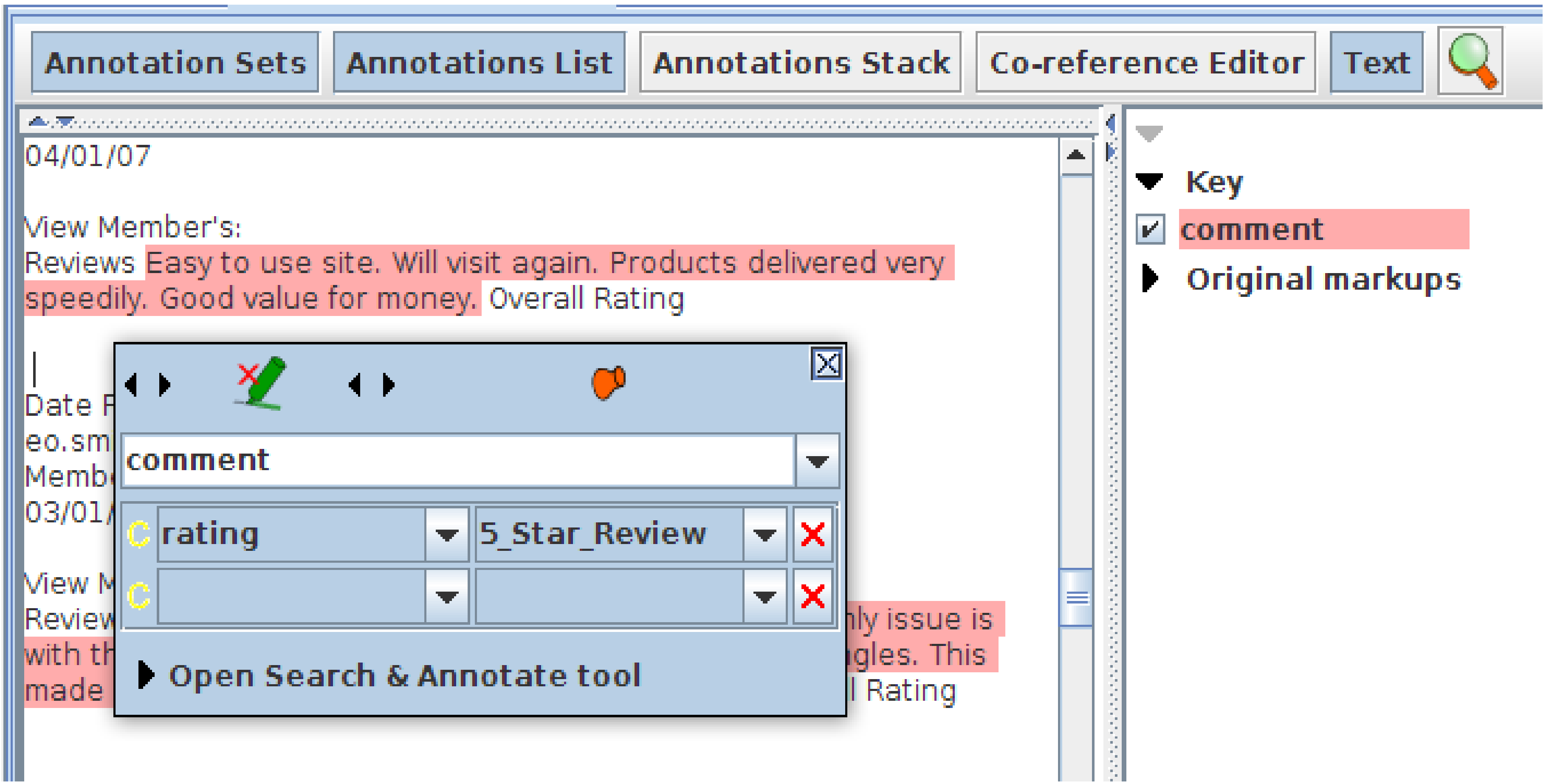
Date Reviewed: 24/01/07

Dbeach135 **Member Since:** 24/01/07 **View Member's:** Reviews	Jessops not only failed to complete the next day delivery, the item sent, a digital picture frame did not meet their specification. We ordered it as they claimed on their website that it accepted XD cards. This however was not the case. Jessops felt that they had done nothing wrong although their website was obviously wrong. This incorrect information still is outstanding and they have done nothing to correct their website even though I have notified them of the error.	**Overall Rating**

Preparing the corpus

- Corpus of 40 documents containing 552 company reviews.
- Each review has a 1- to 5-star rating.
- We pre-processed these in GATE to label each review with a comment annotation with a rating feature (free manual annotation!)
- In ML terms:
    - instance = *comment* annotation
    - class = *rating* feature on the *comment* annotation
    - attributes = NLP features of the underlying text
- We will keep the spans of the comment annotations and use ML to classify them with the *rating* feature

Annotated review



Developing the training application

- We will develop an application that runs a set of NLP components to provide ML instance attributes, and trains the classifier
- Load the ANNIE, Tools, and Learning plugins
- Create a new corpus called "training" and populate it from the directory **ml-exercise/corpora/training** in the hands-on material
- Use a text editor to open the **ml-exercise/paum.xml** config file so we can examine it

Batch learning config (paum.xml)

```
<PARAMETER name="thresholdProbabilityClassification"
           value="0.5"/>
```

- This threshold will probably produce a class for each instance
- Classification problems do not use the other threshold probability parameters

```
<multiClassification2Binary method= "one-vs-others"/>
```

- this is much faster than one-vs-another

```
<ENGINE nickname="PAUM" implementationName="PAUM"
          options=" -p 50 -n 5 -optB 0.0  "/>
```

- Perceptron with uneven margins
- default options

Batch learning config (2)

```
<INSTANCE-TYPE>comment</INSTANCE-TYPE>

<ATTRIBUTE>
  <NAME>Class</NAME>
  <SEMTYPE>NOMINAL</SEMTYPE>
  <TYPE>comment</TYPE>
  <FEATURE>rating</FEATURE>
  <POSITION>0</POSITION>
  <CLASS/></ATTRIBUTE>
```

- Takes comment annotations as instances, and classifies them using the rating feature.
- The classes (values of the rating features) form an unordered set (current limitation of the PR).

Batch learning config (3)

```
<NGRAM>
  <NAME>ngram</NAME>
  <NUMBER>1</NUMBER>
  <CONSNUM>1</CONSNUM>
  <CONS-1>
    <TYPE>Token</TYPE>
    <FEATURE>root</FEATURE>
  </CONS-1>
  </NGRAM>
```

- Uses unigrams of *Token.root* features inside the comment annotations as the instance attributes (bag of words).
- An additional feature in the hands-on file is commented out for you to experiment with later.

Building the training application (1)

- Create the following PRs with the default init parameters:
  - Document Reset PR
  - Annotation Set Transfer
  - ANNIE English Tokeniser
  - ANNIE Sentence Splitter
  - ANNIE POS Tagger
  - GATE Morphological Analyser
- Create a Batch Learning PR with the configFileURL init parameter set to the new location of your **ml-exercise/paum.xml** file
- Create a new Conditional Corpus Pipeline.

Building the application (2)

- We want to copy the comment annotations to the default annotation set to provide the ML instances and classes, but we don't want to remove the Key annotations
- Add the PRs to the pipeline & set some runtime parameters
  - Document Reset:
    - setsToKeep = “Key”
  - Annotation Set Transfer:
    - annotationTypes = empty list (copy all)
    - copyAnnotations = true
    - inputASName = “Key”
    - outputASName & textTagName must be blank

Building the application (3)

- Add the remaining loaded PRs to the pipeline
  - English tokeniser
  - Sentence splitter
  - POS tagger
  - Morphological analyser
  - Batch Learning:
    - inputASName is blank
    - learningMode = TRAINING
- Run it on the training corpus (this should take less than 1 minute)
- The classifier's model is stored in the **savedFiles** directory beside the **paum.xml** file. The model is stored in text files, but they are not meant to be human-readable.

Applying the training model (1)

- Create a "testing" corpus and populate it from the **corpora/testing** directory.
- To apply the classifier, we need to have comment annotations *without* rating features on the default AS. These will give us the instances to classify. A simple JAPE Transducer can do this.
- Load the grammar resources/grammar/copy_comment_spans.jape.
- Insert the grammar in the pipeline after the AS Transfer PR.
- Set the transducer parameters:
  - inputASName = "Key"
  - outputASName = ""

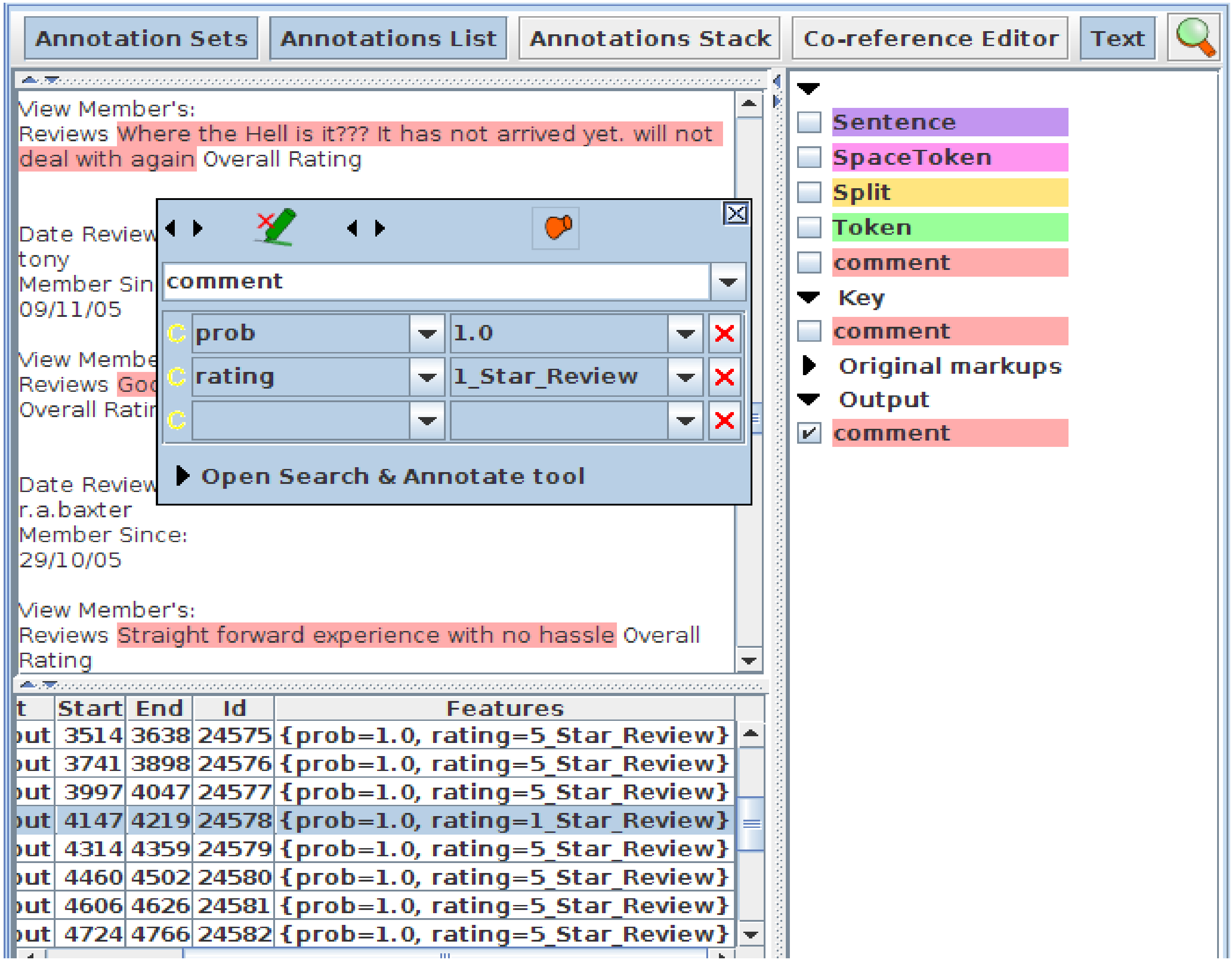
Applying the training model (2)

- Set the AS Transfer PR's run-mode to “no” (red light)
- Set the Batch Learning PR's parameters:
  - inputASName is blank (default AS)
  - learningMode = APPLICATION
  - outputASName = “Output”
- The classifier will get instances (*comment* annotations) and attributes (other annotations' features) from the default AS and put instances with classes (*rating* features) in the Output AS.

Applying the training model (3)

- Run the pipeline on the testing corpus
- Open a few documents and inspect the “comment” annotations:
    - “Key” AS = user ratings (instances and correct classes)
    - default AS = instances & attributes but no classes
    - “Output” AS = instances with classes generated by ML

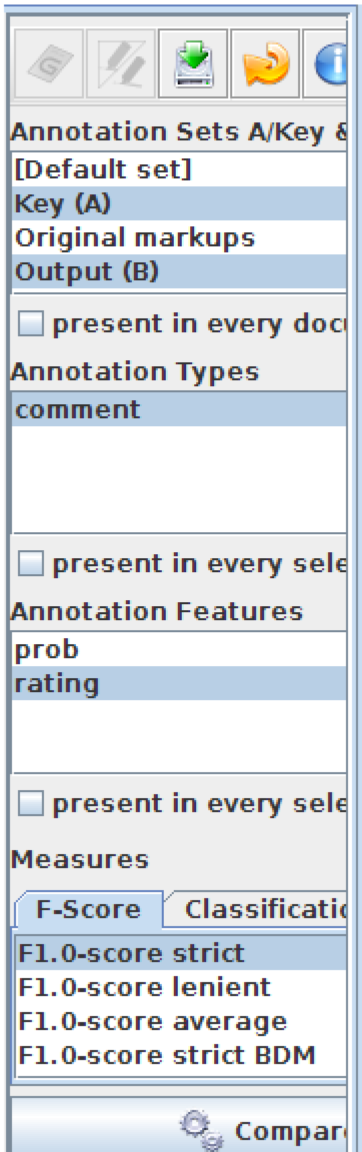
Annotation Results



Evaluation: Precision and Recall

- On the test corpus, click the Corpus Quality Assurance tab.
- Select
  - Annotation Sets A = *Key*, B = *Output*
  - Annotation Types = *comment*
  - Annotation Features = *rating*
  - F-Score = F1.0-score strict
- Click “Compare”
- If every instance has been classified, then the total P = R = F1, because every spurious classification is paired with a missing classification
- Use the “Document statistics” sub-pane of Corpus QA to confirm this, and to see how specific documents were annotated

Corpus QA parameters



Results

Corpus statistics | Document statistics

Annotation	Match	Only A	Only B	Overlap	Rec.B/A	Prec.B/A	F1.0-s.
comment	79	20	20	0	0.80	0.80	0.80
Macro summary					0.80	0.80	0.80
Micro summary	79	20	20	0	0.80	0.80	0.80

Evaluation: Cohen's Kappa

- Click on Measures in the Corpus QA settings and select “Classification”, then “Observed agreement”, and click “Compare”.
- In addition to the document statistics with summary lines, there is now a “Confusion Matrices” tab.
- These matrices show, for the corpus & for each document, how many of each class were classified in that class (or in other classes)
- Cells in these tables can be selected with the mouse (or Ctrl-A to select all) and copied with Ctrl-C, so you can paste them into a spreadsheet.

Built-in cross-validation

- Cross-validation is a standard way to “stretch” the validity of a manually annotated corpus, because it enables you to test on a larger number of documents
- The 5-fold averaged result is more significant than the result obtained by training on 80% of the same corpus and testing on 20% once.
- In GATE, you can't use the Corpus QA tool on the result, but you can get a detailed statistical report at the end, including P, R, & F1 for each class.

Built-in cross-validation

- The config file includes:
  - <EVALUATION method="kfold" runs="5" ratio="0.66" />
  - kfold ignores the ratio setting
  - holdout ignores the runs setting
- The Batch Learning PR will automatically split the corpus into 5 parts, and then
  - train on 1,2,3,4; apply on 5; then
  - train on 1,2,3,5; apply on 4; …
  - train on 2,3,4,5; apply on 1;
  - and average the results.

Built-in cross-validation

- The application can be modified slightly to perform cross-validation according to the config file
- Switch the AS Transfer PR back on (so it copies the comment annoation to the default AS)
- Switch off the JAPE transducer
- Set the Batch Learning PR parameters:
  - inputAS, outputAS = “”
  - learningMode = EVALUATION
- Create a new corpus “all” and populate it from the **corpora/all** directory (all the documents from training and testing.)
- Run the pipeline on the new corpus. This will take a few minutes.
- Click on the Messages pane to view the results

Results

```
*** Averaged results for each label over 5 runs as:

Results of single label:
0 LabelName=1_Star_Review, number of instances=27
  (correct, partialCorrect, spurious, missing)= (1.6, 0.0, 0.6, 5.2);  (precision, recall, F1)= (0.4666667,
0.25333333, 0.3277778);  Lenient: (0.4666667, 0.25333333, 0.3277778)
1 LabelName=2_Star_Review, number of instances=6
  (correct, partialCorrect, spurious, missing)= (0.0, 0.0, 0.0, 1.6);  (precision, recall, F1)= (0.0, 0.0, 0.0);
 Lenient: (0.0, 0.0, 0.0)
2 LabelName=3_Star_Review, number of instances=17
  (correct, partialCorrect, spurious, missing)= (0.8, 0.0, 0.0, 3.6);  (precision, recall, F1)= (0.4,
0.24000001, 0.26666665);  Lenient: (0.4, 0.24000001, 0.26666665)
3 LabelName=4_Star_Review, number of instances=92
  (correct, partialCorrect, spurious, missing)= (1.8, 0.0, 3.4, 21.2);  (precision, recall, F1)=
(0.44666666, 0.0984191, 0.1436075);  Lenient: (0.44666666, 0.0984191, 0.1436075)
4 LabelName=5_Star_Review, number of instances=298
  (correct, partialCorrect, spurious, missing)= (72.8, 0.0, 29.4, 1.8);  (precision, recall, F1)=
(0.71384054, 0.9741704, 0.819043);  Lenient: (0.71384054, 0.9741704, 0.819043)

Overall results as:
  (correct, partialCorrect, spurious, missing)= (77.0, 0.0, 33.4, 33.4);  (precision, recall, F1)=
(0.6971326, 0.6971326, 0.6971326);  Lenient: (0.6971326, 0.6971326, 0.6971326)

This learning session finished!
```

Looking into the future

- Typically, opinion mining looks at social media content to analyse people's explicit opinions about a product or service
- This backwards-looking approach often aims primarily at dealing with problems, e.g. unflattering comments
- A forwards-looking approach aims at looking ahead to understanding potential new needs from consumers
- This is not just about looking at specific comments, e.g. "the product would be better if it had longer battery life", but also about detecting non-specific sentiment
- This is achieved by understanding people's needs and interests in a more general way, e.g. drawing conclusions from their opinions about other products, services and interests.

The problem of sparse data

- One of the difficulties of drawing conclusions from traditional opinion mining techniques is the sparse data issue
- Opinions tend to be based on a very specific product or service, e.g. a particular model of camera, but don't necessarily hold for every model of that brand of camera, or for every product sold by the company
- One solution is figuring out which statements can be generalised to other models/products and which are specific
- Another solution is to leverage sentiment analysis from more generic expressions of motivation, behaviour, emotions and so on, e.g. what type of person buys what kind of camera?

Summary

- Introduced the concept of Opinion Mining and Sentiment Analysis
- Simple examples of rule-based and ML methods for creating OM applications
- Examples of how deeper linguistic information can be useful
- Practice with complex applications
- Looking ahead to the future

More information

- See the following paper for details and evaluation of a more complex version of the twitter application
- D. Maynard and A. Funk. Automatic detection of political opinions in tweets. In Proceedings of MSM 2011: Making Sense of Microposts. Workshop at 8th Extended Semantic Web Conference (ESWC 2011). Heraklion, Greece. June 2011 (download PDF)
- The EU-funded ARCOMEM project is dealing with lots of issues about opinion mining from social media, and uses GATE for this. See http://www.arcomem.eu for more details

Suggestions for further ML experiments...

Suggestions...

- The config file URL is an init parameter, but the contents can be re-loaded, so you can
  - use any text editor on the config file, save the changes, and
  - re-initialize the Batch Learning PR to re-load the file with changes.

Suggestions...

- Try n-grams where n>1
  - Change <NUMBER> in the config
  - Usually this is slower, but sometimes it improves quality
- combining features
  - change <CONSUM> in the config to 2 and uncomment the *Token.orth* element
  - this concatenates the features

Suggestions...

- Adjust the thresholdProbabilityClassification
  - Increasing it may increase precision and decrease recall, and may prevent the classifier from assigning a class to every instance.
  - Decreasing it may increase recall and decrease precision.
  - This is the “pickiness” control of the classifier.

Suggestions...

- Try using other features
  - *Token.string*, *Token.category*, or combinations of these with *Token.root* and *Token.orth*
- You could even include other ANNIE PRs in the pipeline and use Lookup or other annotation types.
  - You need to run the same annotation-creating PRs for training and application.
  - If the config file specifies an annotation that is missing in an instance, the ML PR will throw an exception.