Text Analysis with Machine Learning

Chris DuBois
Dato, Inc.

About me

Chris DuBois

Staff Data Scientist

Ph.D. in Statistics

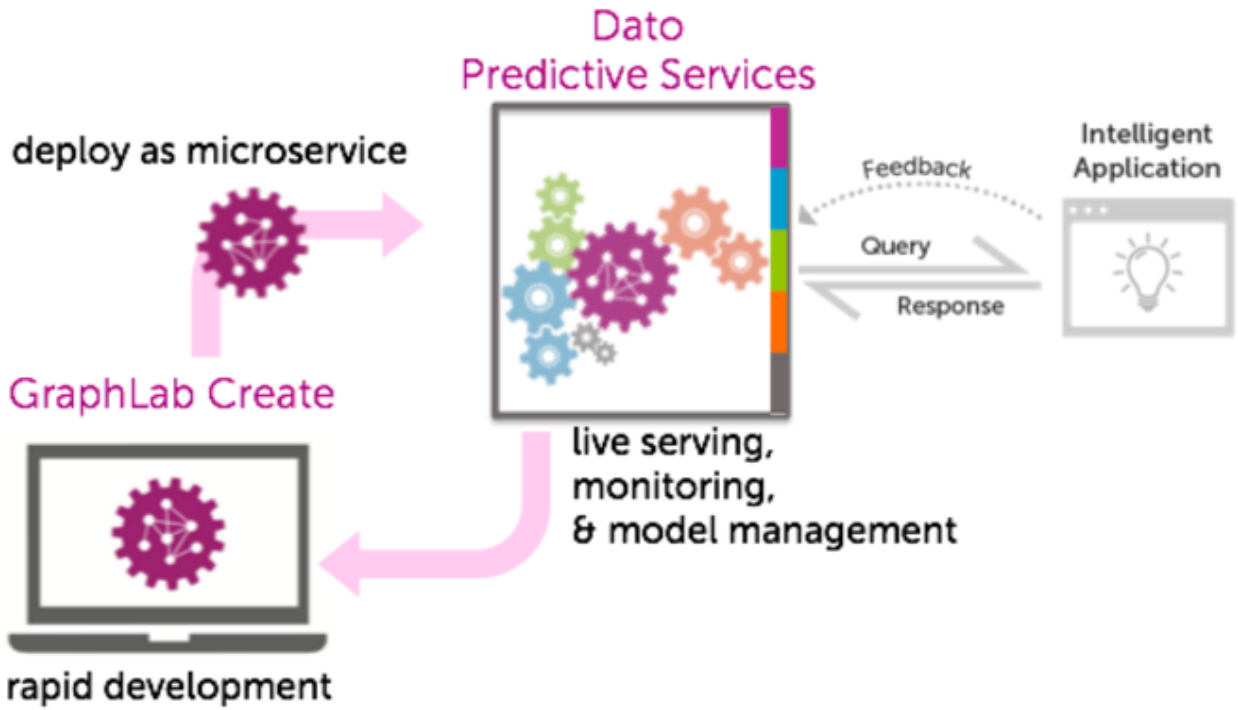
Previously: probabilistic models of social networks and event data occurring over time.

Recently: tools for recommender systems, text analysis, feature engineering and model parameter search.

chris@dato.com

@chrisdubois

About Dato



We aim to help developers…

- develop valuable ML-based applications
- deploy models as intelligent microservices

Quick poll!

Agenda

- Applications of text analysis: examples and a demo
- Fundamentals
  - Text processing
  - Machine learning
  - Task-oriented tools
- Putting it all together
  - Deploying the pipeline for the application
- Quick survey of next steps
  - Advanced modeling techniques

Applications of text analysis

Product reviews and comments

"My faves include juicy pork dumpling (obviously!), noodle with mince pork, and **green beans.**" in 123 reviews

"The **spicy wontons** were our favorite, we didn't particular care for the crab in the soup dumplings and the spinach balanced all of the carbs out." in 74 reviews

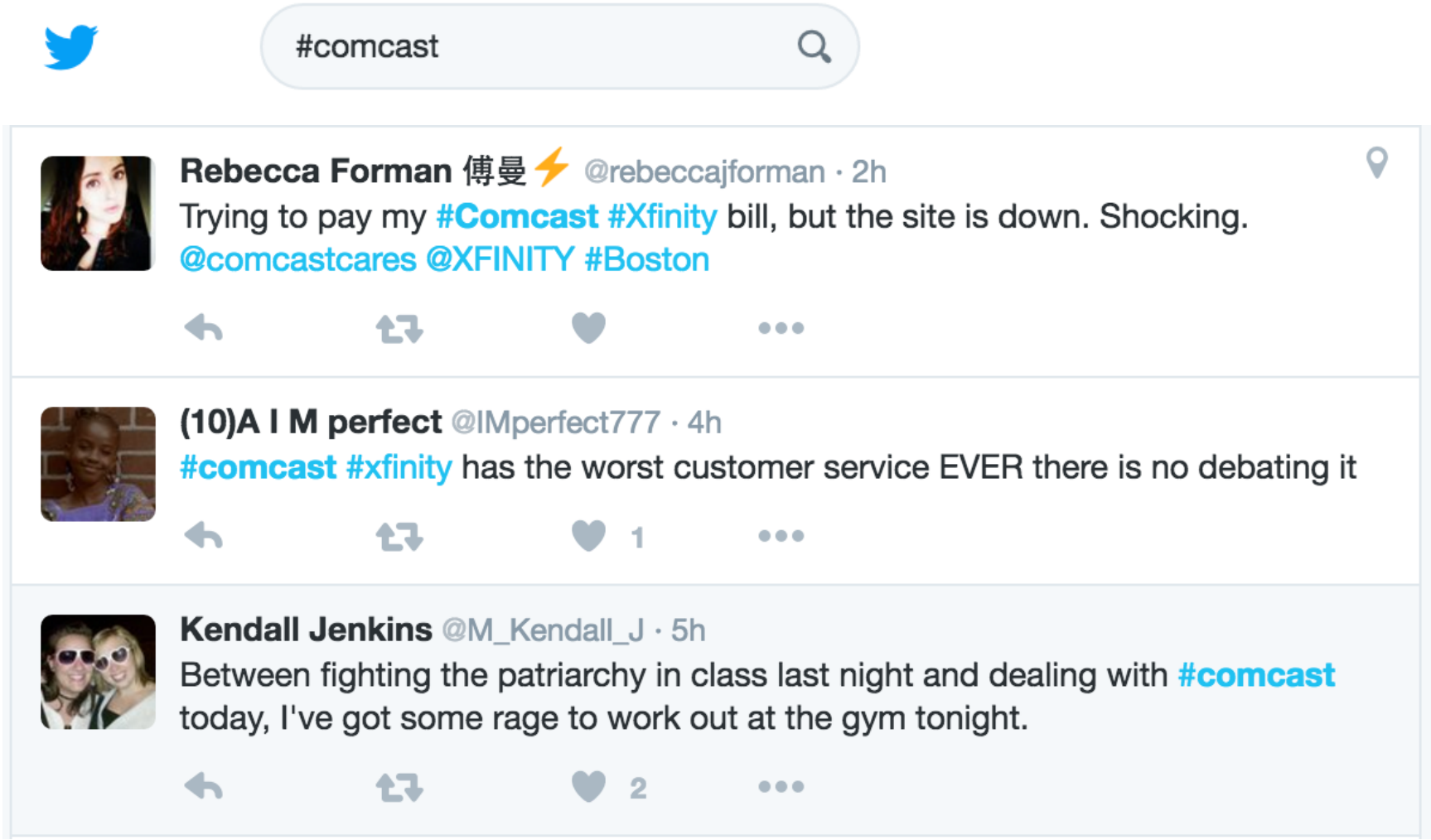
"I gave the **Bellevue location** 4 stars but this new U-Village locale definitely merits 5." in 75 reviews

Show more review highlights

User forums and customer service chats

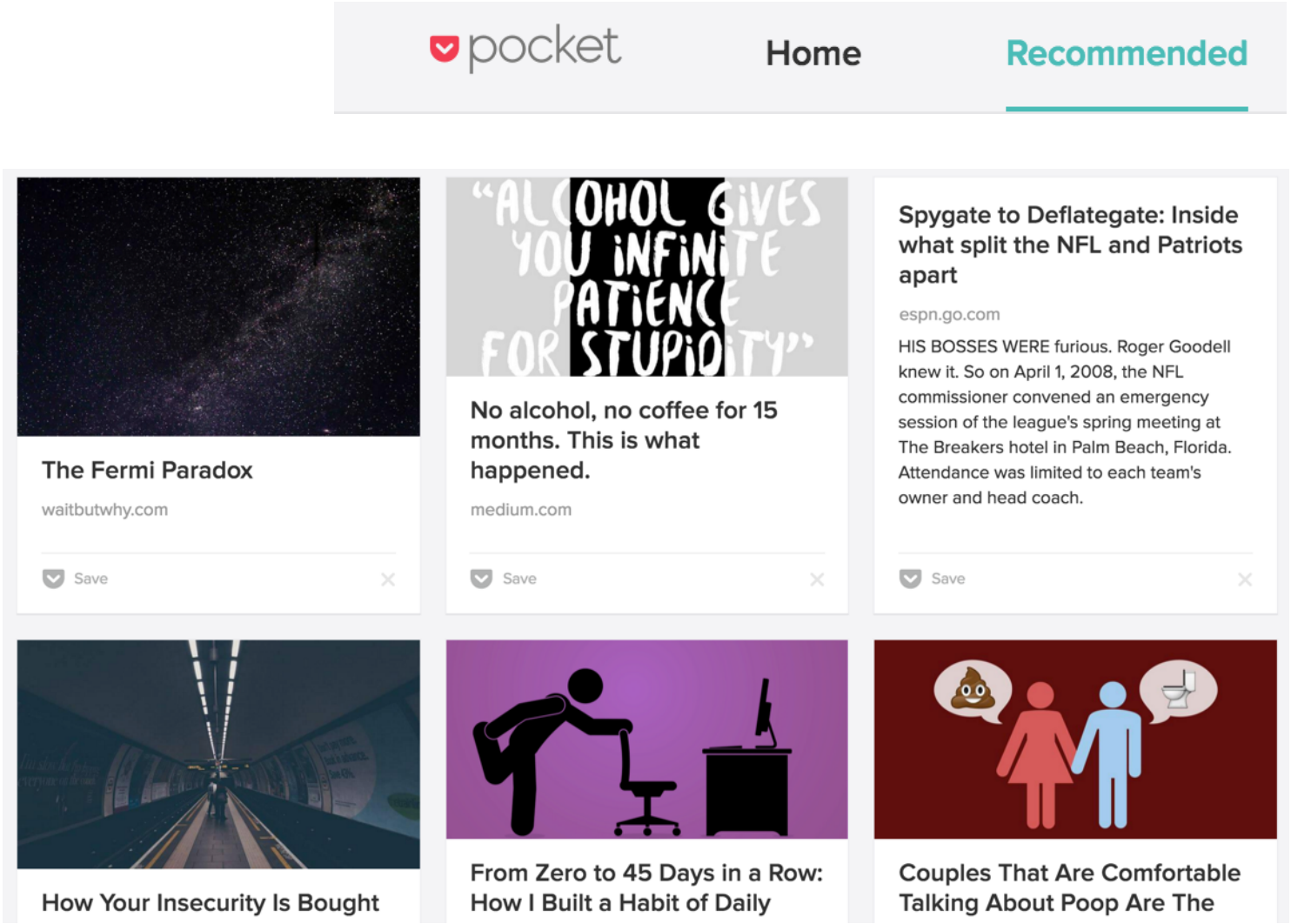
Want to improve Dato products? Participate in a user study! Announcement Started by michaelbrooks January 19 General	22 views	0 comments
Can't import 50mb CSV with graphlab.SFrame.read_csv on EC2 Started by darko 3:34AM Bug Reports	5 views	0 comments
Deep Neural neural: specify the input tensor Most recent by Kevin Noe 12:08AM GraphLab Create	6 views	2 comments
Difference between new_observation_data and new_user_data. Most recent by schawla 12:07AM General	11 views	5 comments
Semi-supervised learning Started by ete 12:02AM General	3 views	0 comments
New features: Partial dependance Most recent by ChrisDuBois April 11 Feature Requests	10 views	1 comment

Social media: blogs, tweets, etc.



Applications of text analysis

News feeds



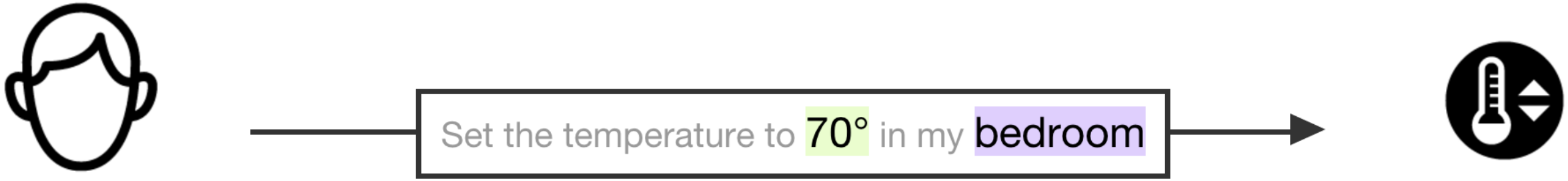
Speech recognition and chat bots

Set the temperature to 70° in my bedroom

Did you try... Remind me to feed the baby tomorrow at 7am

wit.ai

```
intent      = heating control
temperature = 70°F
where       = master_bedroom
```



Applications of text analysis

Aspect mining

Quick poll!

Example of a product sentiment application

Text processing fundamentals

Data

id	*timestamp*	*user_name*	*text*
102	*2016-04-05 9:50:31*	*Bob*	*The food tasted bland and uninspired.*
103	*2016-04-06 3:20:25*	*Charles*	*I would absolutely bring my friends here. I could eat those fries every …*
104	*2016-04-08 11:52*	*Alicia*	*Too expensive for me. Even the water was too expensive.*
…	…	…	…
int	*datetime*	*str*	*str*

Transforming string columns

id	*text*
102	*The food tasted bland and uninspired.*
103	*I would absolutely bring my friends here. I could eat those fries every …*
104	*Too expensive for me. Even the water was too expensive.*
…	…
int	*str*

Tokenizer
CountWords
NGramCounter
TFIDF
RemoveRareWords
SentenceSplitter
ExtractPartsOfSpeech

stack/unstack
Pipelines

Tokenization

Convert each document into a list of tokens.

INPUT `“The caesar salad was amazing.”`

OUTPUT `[“The”, “caesar”, “salad”, “was”, “amazing.”]`

Bag of words representation

Compute the number of times each word occurs.

INPUT `“The ice cream was the best.”`

OUTPUT

```
{
 “the”: 2,
 “ice”: 1,
 “cream”: 1,
 “was”: 1,
 “best”: 1
}
```

N-Grams (words)

Compute the number of times each set of words occur.

INPUT “Give it away, give it away, give it away now”

OUTPUT
```
{
   “give it”:   3,
   “it away”:   3,
   “away give”: 2,
   “away now”:  1
}
```

N-Grams (characters)

Compute the number of times each set of characters occur.

INPUT `“mississippi”`

OUTPUT
```
{
   “mis”: 1,
   “iss”: 2,
   “sis”: 1,
   “ssi”: 2,
   “sip”: 1,
   ...
}
```

TF-IDF representation

Rescale word counts to discriminate between common and distinctive words.

INPUT `{“this”: 1, “is”: 1, “a”: 2, “example”: 3}`

OUTPUT `{“this”: .3, “is”: .1, “a”: .2, “example”: .9}`

Low scores for words that are common among all documents
High scores for rare words occurring often in a document

Remove rare words

Remove rare words (or pre-defined stopwords).

INPUT `“I like green eggs3 and ham.”`

OUTPUT `“I like green and ham.”`

Split by sentence

Convert each document into a list of sentences.

INPUT `“It was delicious. I will be back.”`

OUTPUT `[“It was delicious.”, “I will be back.”]`

Extract parts of speech

Identify particular parts of speech.

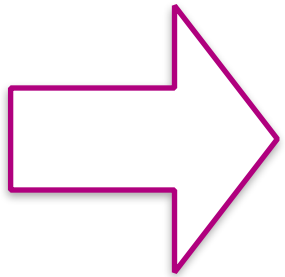
INPUT `“It was delicious. I will go back.”`

OUTPUT `[“delicious”]`

stack

Rearrange the data set to “stack” the list column.

id	text
102	[“The food was great.”, “I will be back.”]
103	[“I would absolutely bring my friends here.”, “I could eat those fries every day.”]
104	[“Too expensive for me.”]
…	…
int	list

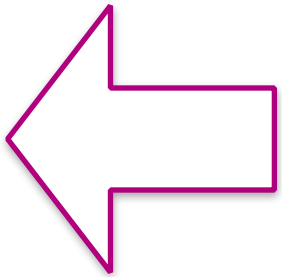


id	text
102	“The food was great.”
102	“I will be back.”
103	“I would absolutely bring my friends here.”
…	…
int	str

unstack

Rearrange the data set to “unstack” the string column.

id	text
102	[“The food was great.”, “I will be back.”]
103	[“I would absolutely bring my friends here.”, “I could eat those fries every day.”]
104	[“Too expensive for me.”]
…	…
int	list



id	text
102	“The food was great.”
102	“I will be back.”
103	“I would absolutely bring my friends here.”
…	…
int	str

Pipelines

Combine multiple transformations to create a single pipeline.

```
from graphlab.feature_engineering import
   WordCounter, RareWordTrimmer

f = gl.feature_engineering.create(sf,
         [WordCounter, RareWordTrimmer])
f.fit(data)
f.transform(new_data)
```

Machine learning toolkits

Task-oriented toolkits and building blocks

Autotagger = matches unstructured text queries to a set of strings

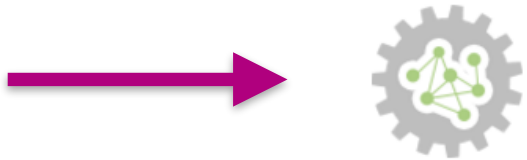
= Nearest neighbors + NGramCounter

Nearest neighbors

Reference

id	*TF-IDF of text*
int	*dict*

`NearestNeighborsModel`



Query

id	*TF-IDF of text*
int	*dict*

Result

query_id	*reference_id*	*score*	*rank*
int	*int*	*float*	*int*

Autotagger = Nearest Neighbors + Feature Engineering

Reference

id	char	...
int	dict	...

`AutotaggerModel`

Query

id	char ngrams	...
int	dict	

Result

query_id	reference_id	score	rank
int	int	float	int

Task-oriented toolkits and building blocks

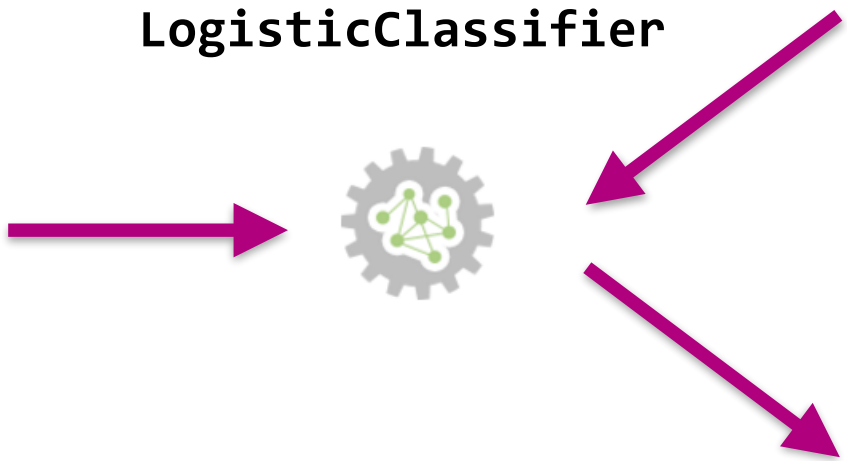
SentimentAnalysis = predicts positive/ negative sentiment in unstructured text

= WordCounter + Logistic Classifier

Logistic classifier

Training data

label	*feature(s)*
int/str	*int/float/str/dict/list*



New data

label	*feature(s)*
int/str	*int/float/str/dict/list*

scores
float

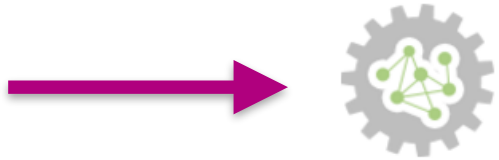
Probability label=1

Sentiment analysis = LogisticClassifier + Feature Engineering

Training data

label	*bag of words*
int/str	*dict*

`SentimentAnalysis`



New data

label	*bag of words*
int/str	*dict*

scores
float

Sentiment scores (probability label = 1)

Product sentiment toolkit

```
>>> m = gl.product_sentiment.create(sf, features=['Description'], splitby='sentence')
>>> m.sentiment_summary(['billing', 'cable', 'cost', 'late', 'charges', 'slow'])
```

keyword	sd_sentiment	mean_sentiment	review_count
cable	0.302471264675	0.285512408978	1618
slow	0.282117103769	0.243490314737	369
cost	0.283310577512	0.197087019219	291
charges	0.164350792173	0.0853637431588	1412
late	0.119163914305	0.0712757752753	2202
billing	0.159655783707	0.0697454360245	583

Code for deploying the demo

Advanced topics in modeling text data

Topic models

Cluster words according to their co-occurrence in documents.

The burrito was terrible. I…

Sometimes sushi here …

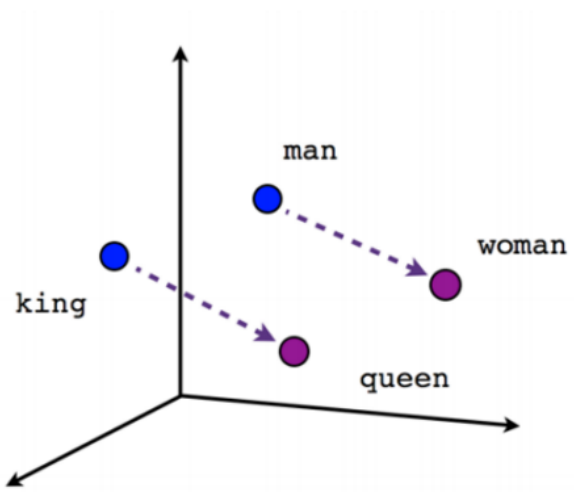
The waiters never came until…

When you need gyoza, you…

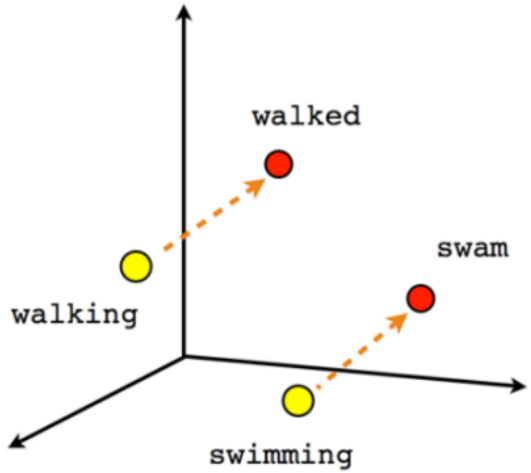
My favorite place ever! You…

Word2vec

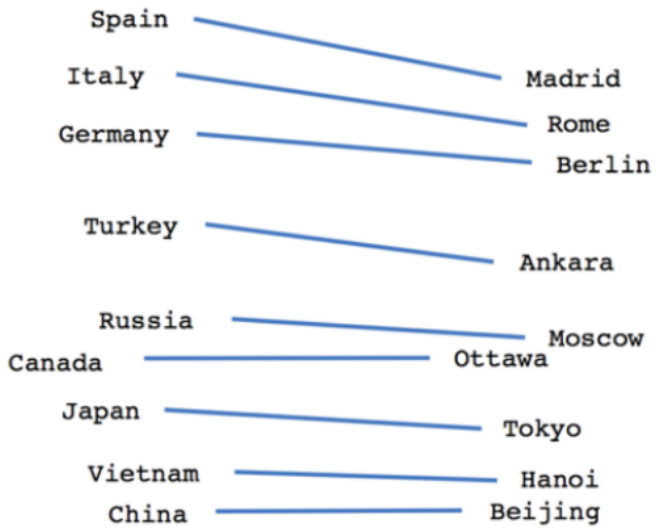
Embeddings where nearby words are used in similar contexts.



Male-Female



Verb tense



Country-Capital

Image: https://www.tensorflow.org/versions/r0.7/tutorials/word2vec/index.html

LSTM (Long-Short Term Memory)

Train a neural network to predict the next word given previous words.

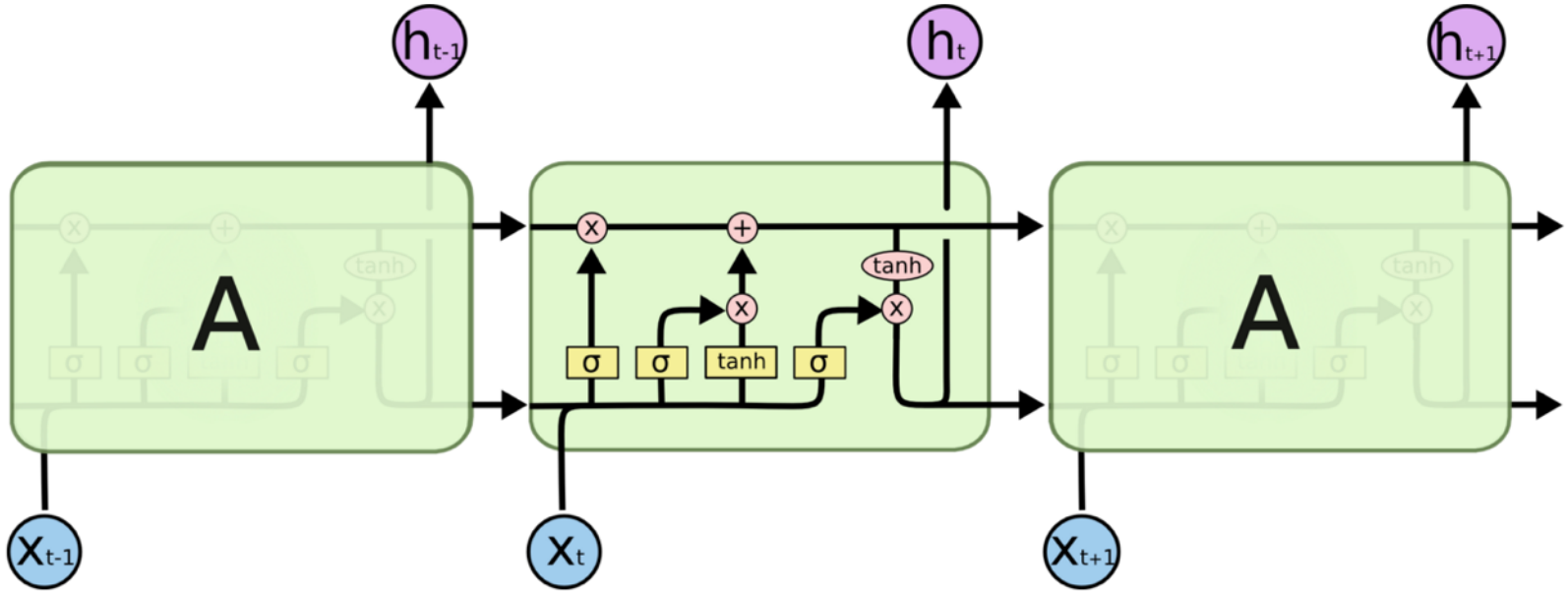


Image: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Conclusion

- Applications of text analysis: examples and a demo
- Fundamentals
  - Text processing
  - Machine learning
  - Task-oriented tools
- Putting it all together
  - Deploying the pipeline for the application
- Quick survey of next steps
  - Advanced modeling techniques

Next steps

Webinar material?	Watch for an upcoming email!
Help getting started?	Userguide, API docs, Forum
More webinars?	Benchmarking ML, data mining, and more

Contact: Chris DuBois, chris@dato.com

Python tools for text analysis

Unifying text analysis

Parsing
(tokens, sentences, parts of speech)

nltk
TextBlob
spaCy

Feature engineering
(ngrams, tf-idf)

scikit learn

Pre-trained models

AYLIEN
MonkeyLearn
wit.ai

Topic models

gensim

RNNs, LSTMs

mxnet
TensorFlow™