# Data Mining for Business Analytics

## Lecture 9: Representing and Mining Text

**Stern School of Business**
**New York University**
**Spring 2014**

# Dealing with Text

- Data are represented in ways natural to problems from which they were derived
- Vast amount of text..
- If we want to apply the many data mining tools that we have at our disposal, we must
  - either engineer the data representation to match the tools (representation engineering), or
  - build new tools to match the data

# Why Text is Difficult

- Text is “unstructured”
  - Linguistic structure is intended for human communication and not computers
- Word order matters sometimes
- Text can be dirty
  - People write ungrammatically, misspell words, abbreviate unpredictably, and punctuate randomly
  - Synonyms, homograms, abbreviations, etc.
- Context matters

# Text Representation

- **Goal**: Take a set of documents –each of which is a relatively free-form sequence of words– and turn it into our familiar feature-vector form
- A collection of documents is called a *corpus*
- *A document is composed of individual tokens or terms*
- *Each document is one instance*
  - *but we don't know in advance what the features will be*

# “Bag of Words”

- Treat every document as just a collection of individual words
  - Ignore grammar, word order, sentence structure, and (usually) punctuation
  - Treat every word in a document as a potentially important keyword of the document
- What will be the feature’s value in a given document?
  - Each document is represented by a one (if the token is present in the document) or a zero (the token is not present in the document)
- Straightforward representation
- Inexpensive to generate
- Tends to work well for many tasks

# Pre-processing of Text

The following steps should be performed:

- The case should be normalized
  - Every term is in lowercase
- Words should be stemmed
  - Suffixes are removed
  - E.g., noun plurals are transformed to singular forms
- Stop-words should be removed
  - A stop-word is a very common word in English (or whatever language is being parsed)
  - Typical words such as the words *the*, *and*, *of*, and *on* are removed

# Term Frequency

- Use the word count (frequency) in the document instead of just a zero or one
  - Differentiates between how many times a word is used

# Normalized Term Frequency

- Documents of various lengths
- Words of different frequencies
  - Words should not be *too common* or *too rare*
  - Both upper and lower limit on the number (or fraction) of documents in which a word may occur
  - Feature selection is often employed
- The raw term frequencies are normalized in some way,
  - such as by dividing each by the total number of words in the document
  - or the frequency of the specific term in the corpus

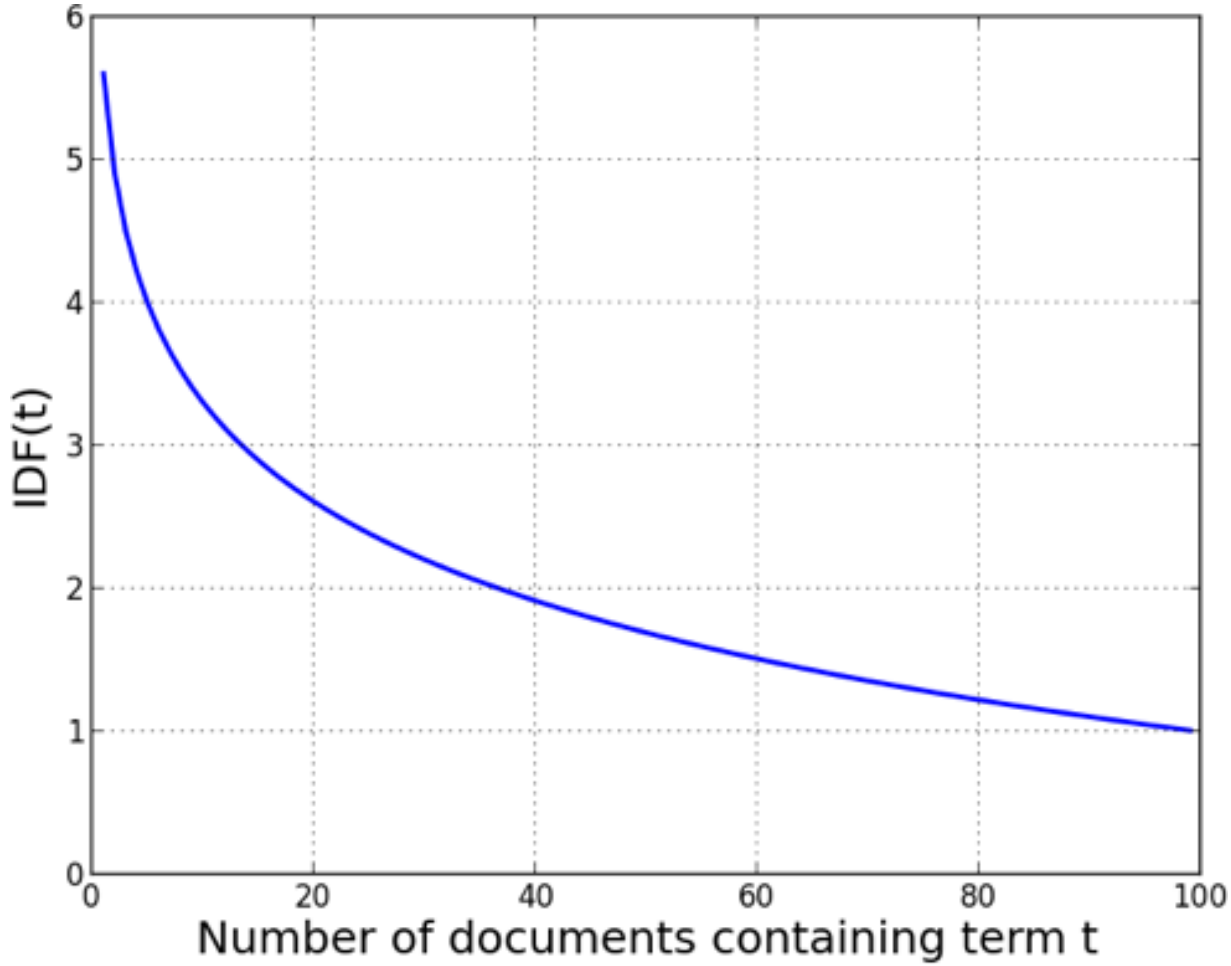
# TF-IDF

$$\mathrm{TFIDF}(t, d) = \mathrm{TF}(t, d) \times \mathrm{IDF}(t)$$

- Inverse Document Frequency (IDF) of a term

$$\mathrm{IDF}(t) = 1 + \log\left(\frac{\text{Total number of documents}}{\text{Number of documents containing } t}\right)$$

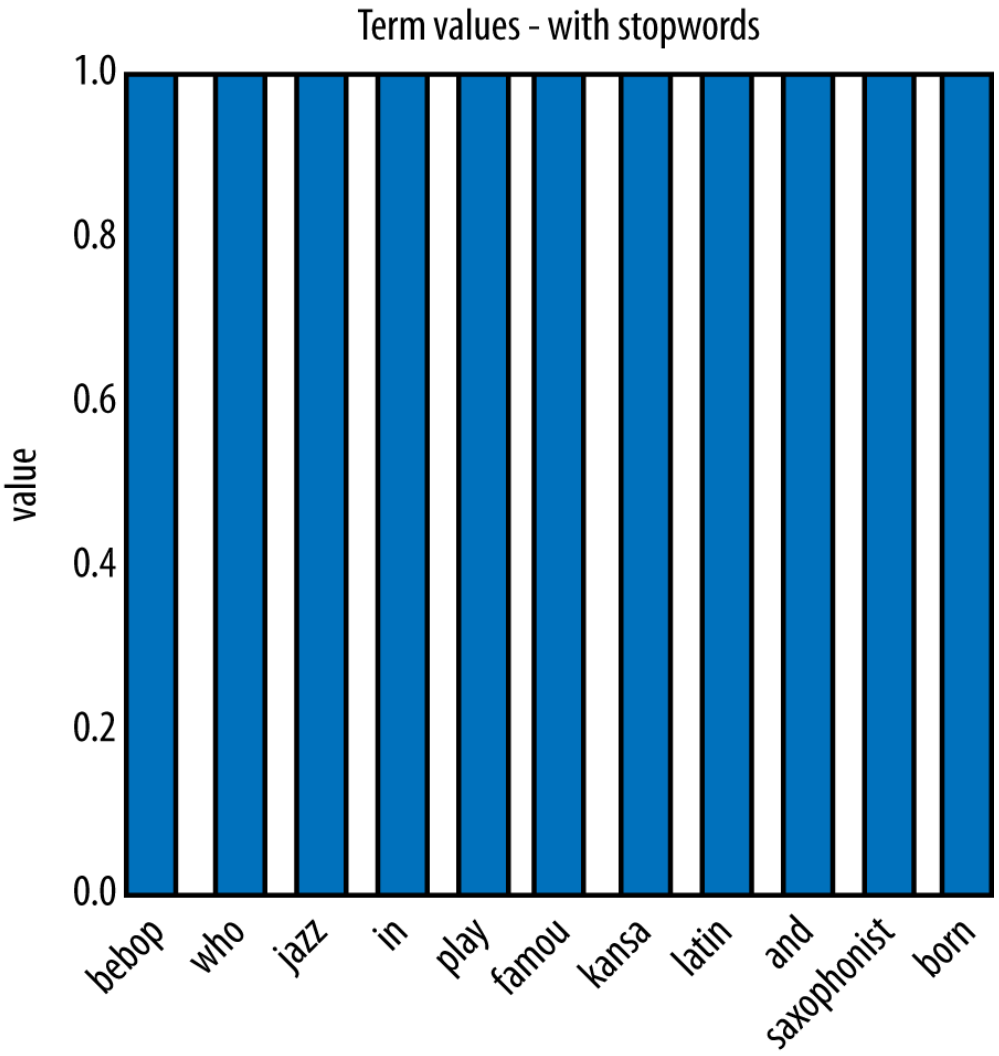
# TFIDF


IDF(t)
Number of documents containing term t
0
1
2
3
4
5
6
0
20
40
60
80
100

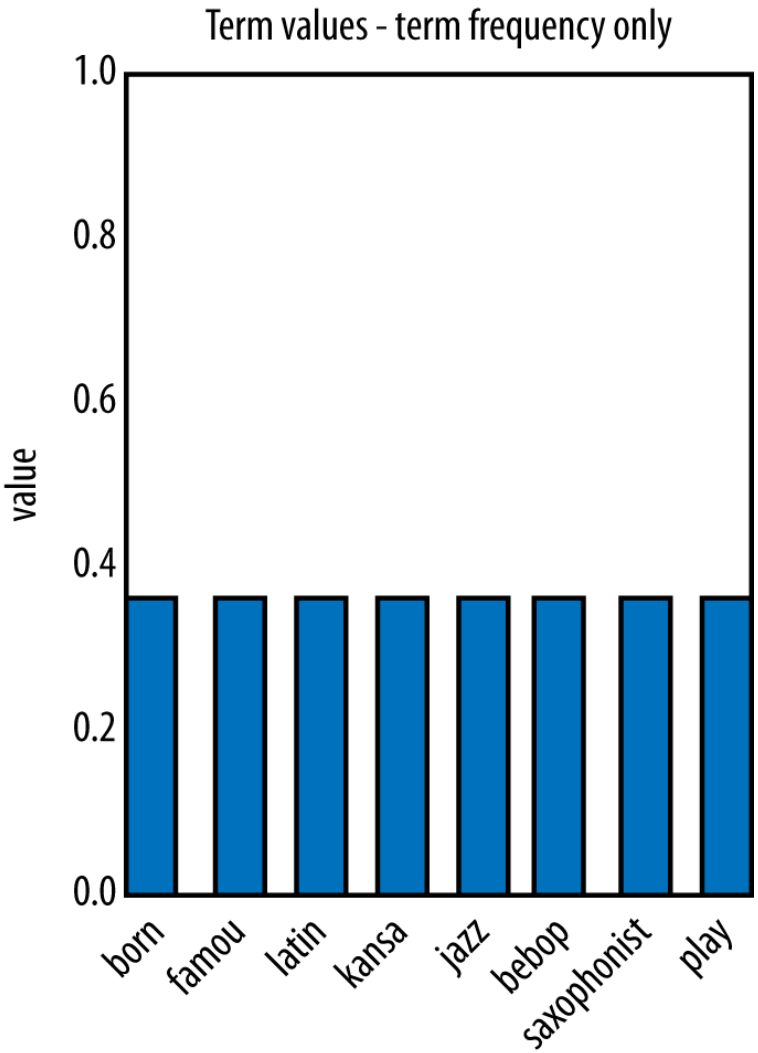
# Example: Jazz Musicians

- 16 prominent jazz musicians and excerpts of their biographies from Wikipedia
- Nearly 2,000 features after stemming and stop-word removal!
- Consider the sample phrase “Famous jazz saxophonist born in Kansas who played bebop and latin”

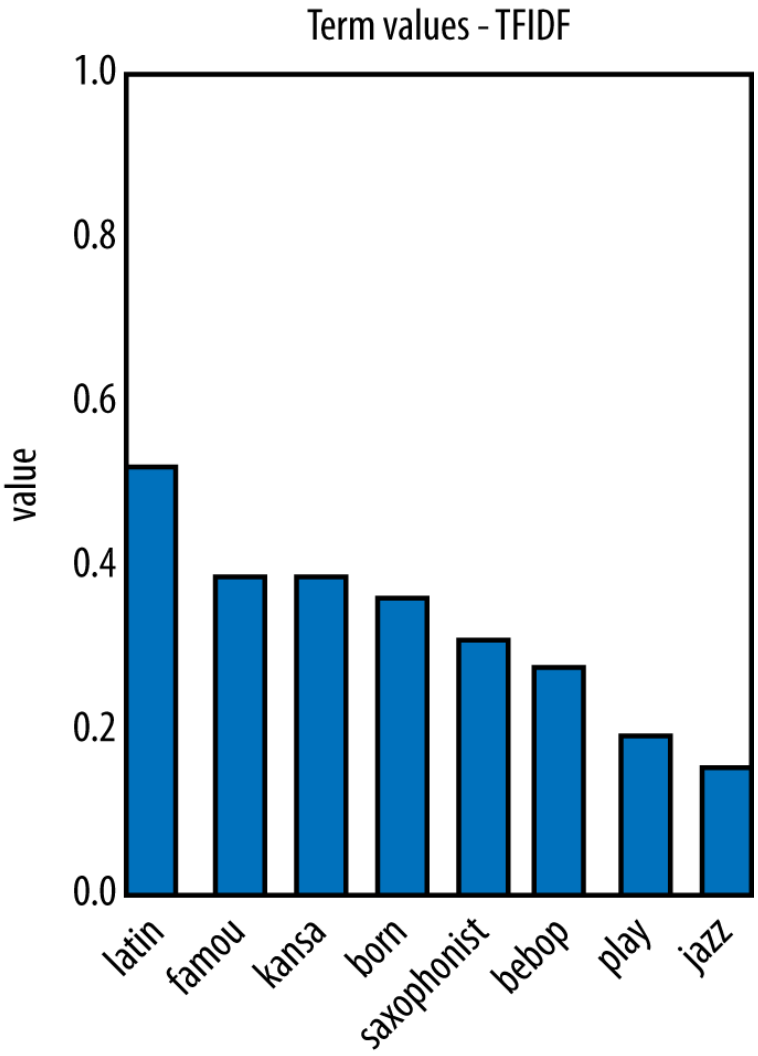
# Example: Jazz Musicians



# Example: Jazz Musicians



# Example: Jazz Musicians



# Example: Jazz Musicians

| Musician | Similarity | Musician | Similarity |
|---|---|---|---|
| Charlie Parker | 0.135 | Count Basie | 0.119 |
| Dizzie Gillespie | 0.086 | John Coltrane | 0.079 |
| Art Tatum | 0.050 | Miles Davis | 0.050 |
| Clark Terry | 0.047 | Sun Ra | 0.030 |
| Dave Brubeck | 0.027 | Nina Simone | 0.026 |
| Thelonius Monk | 0.025 | Fats Waller | 0.020 |
| Charles Mingus | 0.019 | Duke Ellington | 0.017 |
| Benny Goodman | 0.016 | Louis Armstrong | 0.012 |

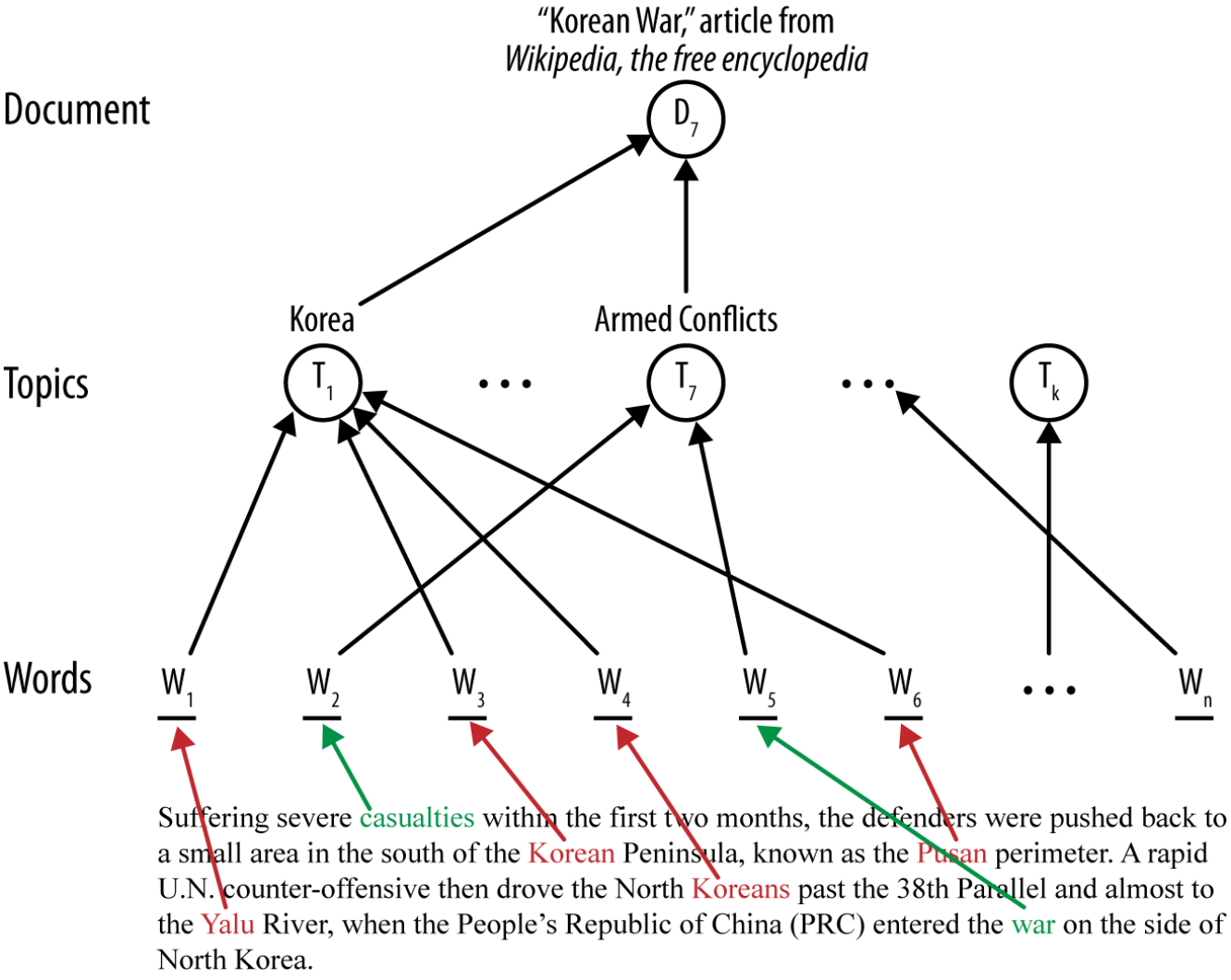
# Beyond “Bag of Words”

- $N$-gram Sequences
- Named Entity Extraction
- Topic Models

# N-gram Sequences

- In some cases, **word order is important** and you want to preserve some information about it in the representation
- A next step up in complexity is to include sequences of adjacent words as terms
- Adjacent pairs are commonly called bi-grams
- Example: “The quick brown fox jumps”
  - It would be transformed into {quick, brown, fox, jumps, quick_brown, brown_fox, fox_jumps}
- N-grams they greatly increase the size of the feature set

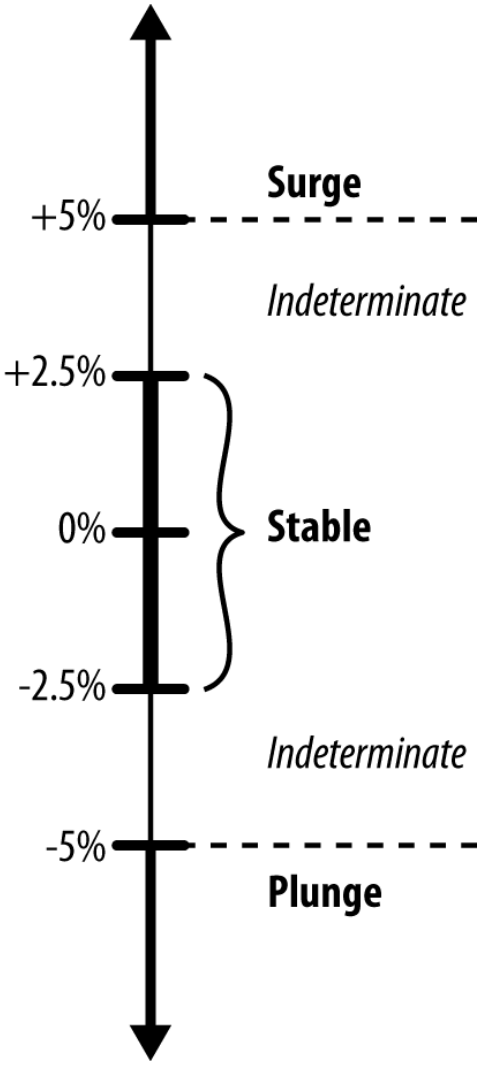
# Topic Models



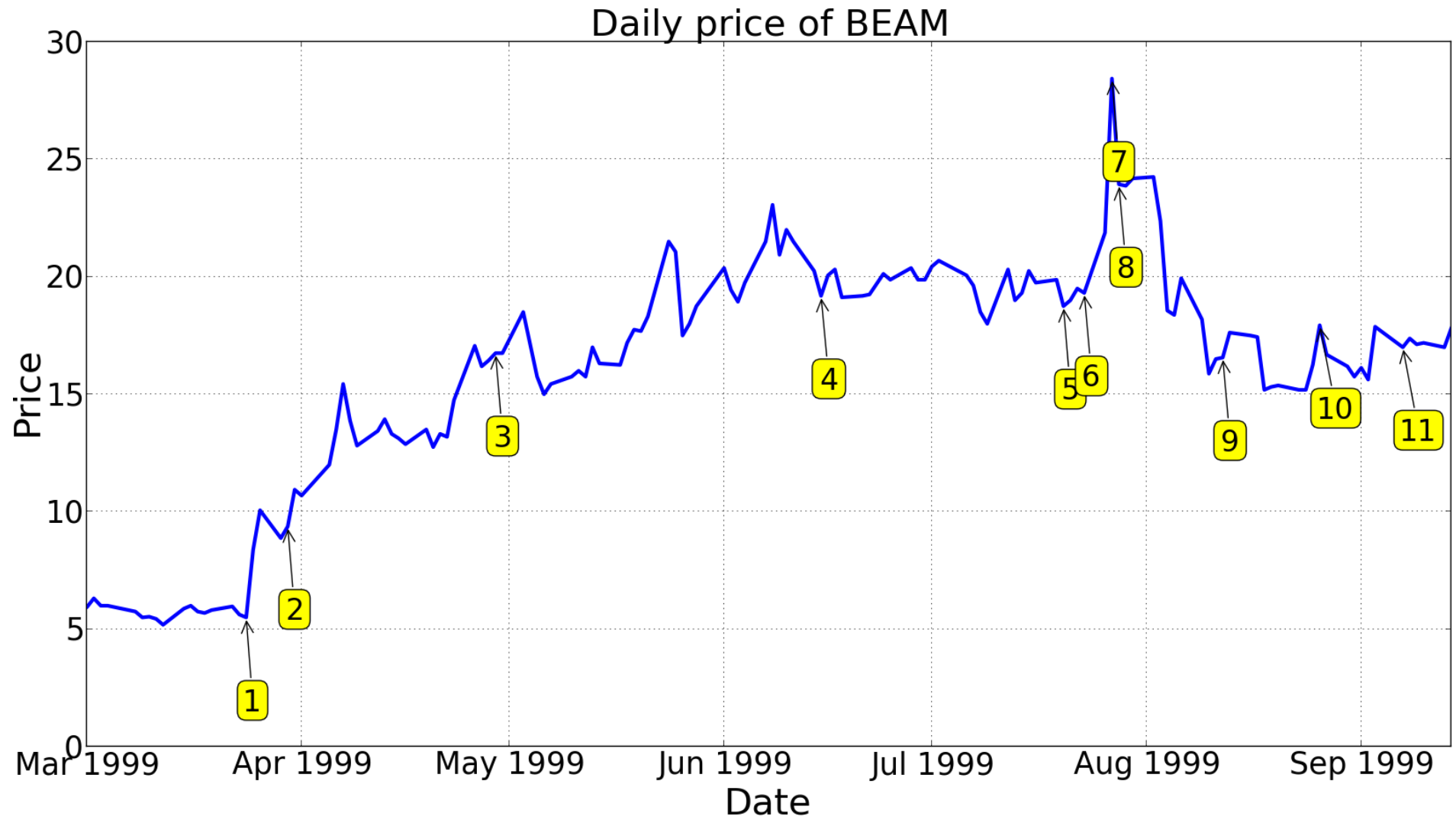
# Text Mining Example

**Task**: predict the stock market based on the stories that appear on the news wires

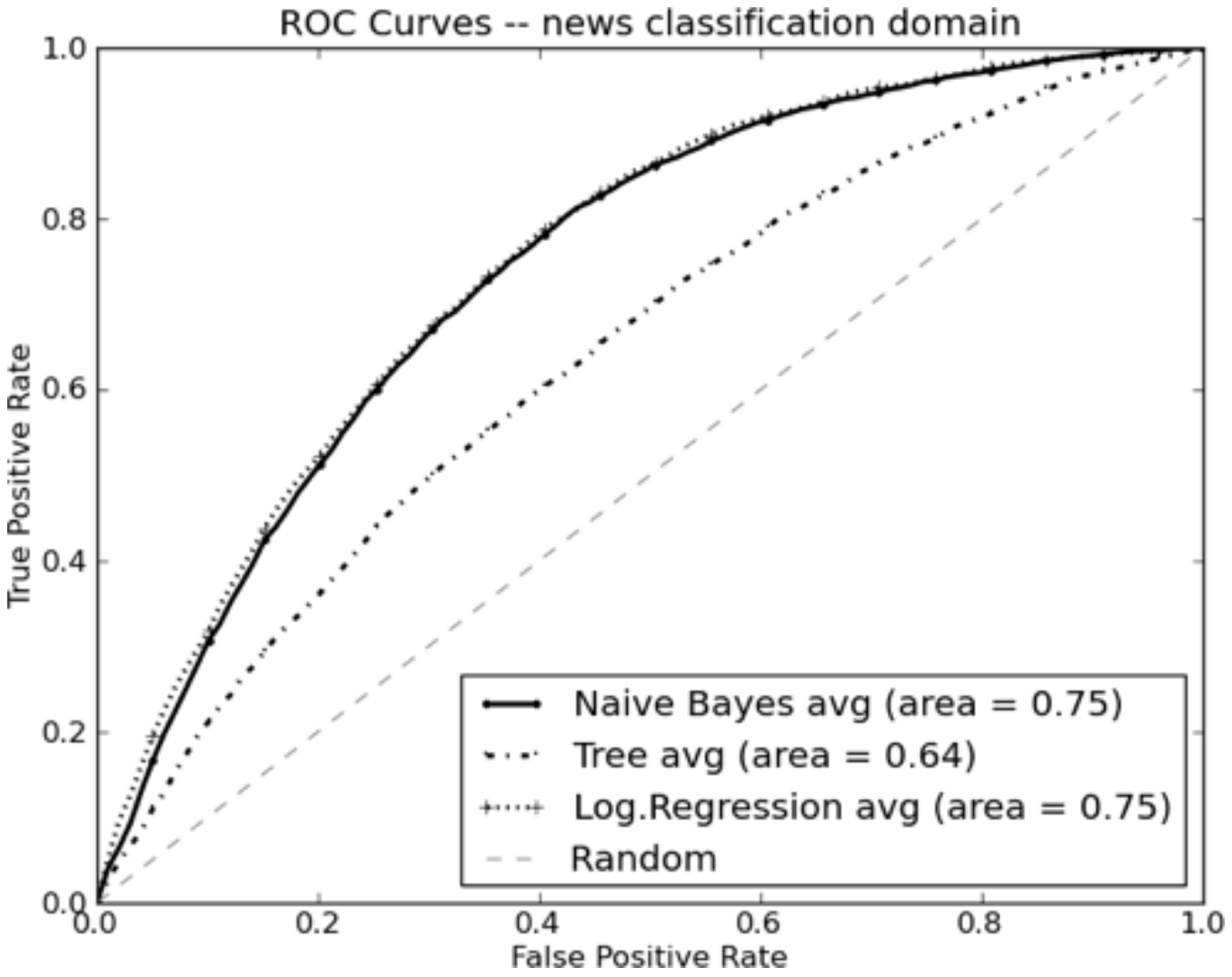
# Mining News Stories to Predict Stock Price Movement



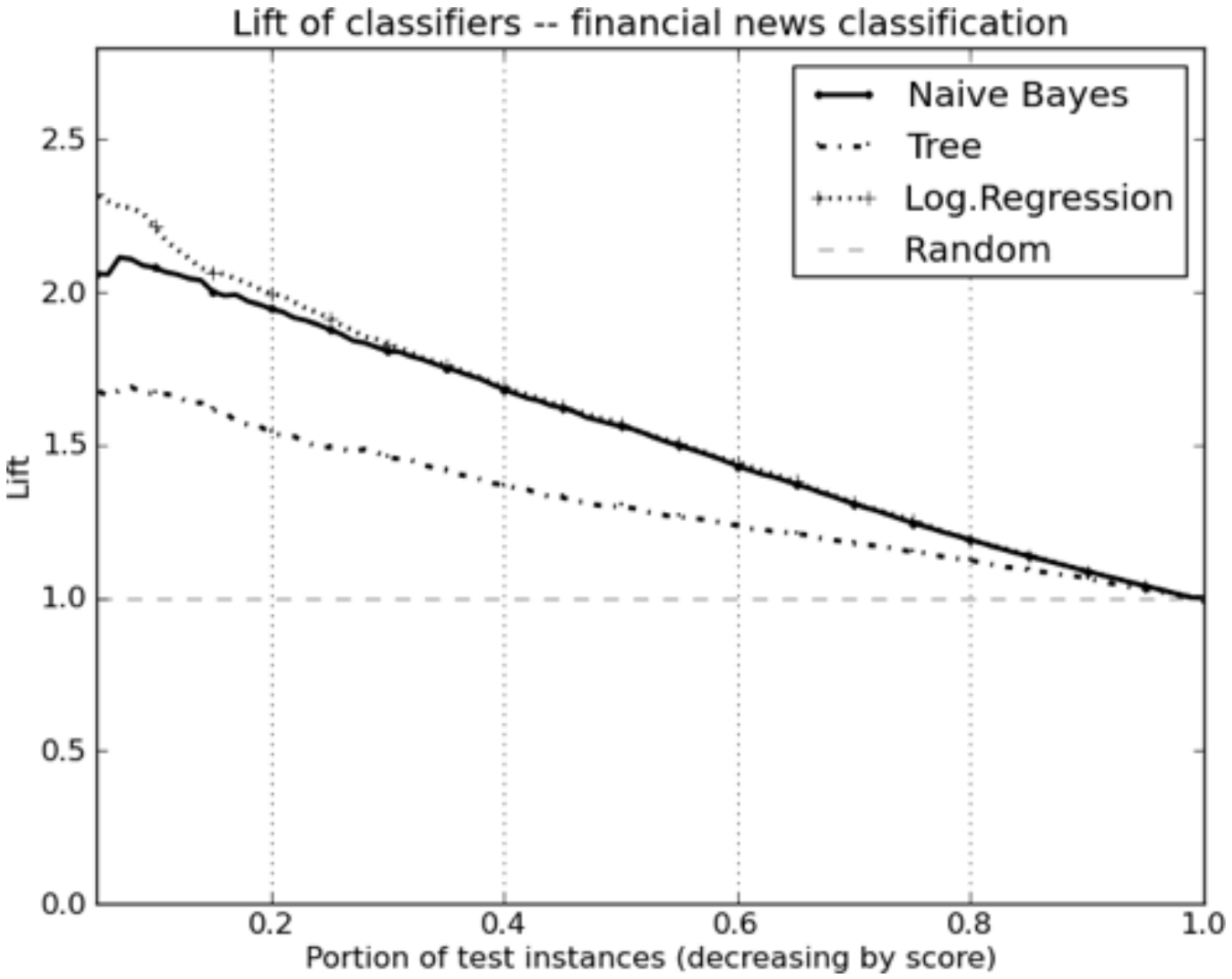
# Mining News Stories to Predict Stock Price Movement



# Mining News Stories to Predict Stock Price Movement



# Mining News Stories to Predict Stock Price Movement



# Mining News Stories to Predict Stock Price Movement

alert(s,ed), architecture, auction(s,ed,ing,eers), average(s,d), award(s,ed), bond(s), brokerage, climb(ed,s,ing), close(d,s), comment(ator,ed,ing,s), commerce(s), corporate, crack(s,ed,ing), cumulative, deal(s), dealing(s), deflect(ed,ing), delays, depart(s,ed), department(s), design(ers,ing), economy, econtent, edesign, eoperate, esource, event(s), exchange(s), extens(ion,ive), facilit(y,ies), gain(ed,s,ing), higher, hit(s), imbalance(s), index, issue(s,d), late(ly), law(s,ful), lead(s,ing), legal(ity,ly), lose, majority, merg(ing,ed,es), move(s,d), online, outperform(s,ance,ed), partner(s), payments, percent, pharmaceutical(s), price(d), primary, recover(ed,s), redirect(ed,ion), stakeholder(s), stock(s), violat(ing,ion,ors)

# Thanks!

# Questions?