*Лидия Пивоварова*

# Кластеризация документов

На основе статьи:

Nicholas O. Andrews and Edward A. Fox, Recent Developments in Document Clustering, October 16, 2007

http://eprints.cs.vt.edu/archive/00001000/01/docclust.pdf

дополнение из книги

"**Introduction to data mining, by Tan, Steinbach, and Kumar**"

# Disclaimer

- Очень много математики. Я не со всем до конца разобралась
- Зато я развернула некоторые базовые понятия, которые в статье даются без пояснений
- Плохая новость: 42 слайда
- Хорошая новость: я не собираюсь подробно обсуждать все

# 1. Введение

Кластеризация документов (далее – просто кластеризация) – это процесс обнаружения естественных групп в коллекции документов.
Текст обзорный, он упоминает основные **классы** алгоритмов и описывает по два-три самых распространенных алгоритма из каждого класса.

# Основные типы кластеризации

- Восходящая/нисходящая кластеризации (hierarcical / partitional)
- Исключающая, перекрывающая и нечеткая кластеризации (exclusive / overlapping)
- Полная и частичная кластеризации (complete / partial clustering)

# Восходящая/нисходящая кластеризация

- Иерархическая кластеризация (восходящая) - допускаем наличие подкластеров, осуществляется в несколько приемов, в результате образуется в иерархическое дерево (дендрограмму).
- Нисходящая (плоская) кластеризация - предполагает разделение на кластеры сразу, причем один объект относится только к одному кластеру.

# Исключающая, перекрывающая и нечеткая кластеризации (exclusive / overlapping/ fuzzing)

- Исключающая – каждый объект может быть отнесен только к одному кластеру
- Перекрывающая - используется, если объект принадлежит к нескольким группам или находится между двумя кластерами.
- Нечеткая или вероятностные кластеризации являются частными случаями перекрывающей кластеризации. Тогда каждый объект относится к кластеру с определенным весом или вероятностью. Например, вес от 0 до1, где 0 – абсолютно не принадлежит, 1 – полностью принадлежит.

# Полная и частичная кластеризации (complete/ partial)

- Метод полной кластеризации - каждый объект обязательно относится к кластеру
- Частичная кластеризация –некоторые объекты не принадлежат к четко определенным группам, поскольку могут являться выбросами, шумами и т.п.

# Определение расстояние между элементами

- Вычисление Евклидова расстояния (если известны координаты точек в пространстве)

$$D(x_i; x_j) = \sqrt{\sum_i^n (x_i - x_j)^2}$$

- Квадрат Евклидова расстояния
- Манхэттенское расстояние (дает те же результаты, что и Евклидово расстояние, но влияние отдельных выбросов уменьшается):

$$D(x_i; x_j) = \sum |x_i - x_j|$$

- Расстояние Чебышева
  Используется, когда нужно определить два объекта как «различные», если они различаются по какой-либо одной координате

$$D(x_i; x_j) = \max(|x_i - x_j|)$$

# Методы объединения кластеров

- **Метод ближнего соседа** или **одиночная связь**. Расстояние между двумя кластерами определяется расстоянием между двумя наиболее близкими объектами в различных кластерах.
- **Метод наиболее удаленных соседей**. Расстояния между кластерами определяются наибольшим расстоянием между любыми двумя объектами в различных кластерах
- **Метод Варда.** Расстояние как прирост суммы квадратов расстояний объектов до центров кластеров, получаемый в результате их объединения
- **Метод невзвешенного попарного среднего.** В качестве расстояния между двумя кластерами берется среднее расстояние между всеми парами объектов в них.
- **Метод взвешенного попарного среднего.** Метод похож на метод невзвешенного попарного среднего, но в качестве весового коэффициента используется размер кластера
- **Невзвешенный центроидный метод.** В качестве расстояния между двумя кластерами в этом методе берется расстояние между их центрами тяжести.
- **Взвешенный центроидный метод.** Метод похож на предыдущий, но для учета разницы между размерами кластеров (числе объектов

# 2. Оценка качества кластеризации

- Не существует единого (общепризнанного, применимого во всех случаях) метода оценки
- Оценка предполагает, что коллекция (или часть коллекции) размечена человеком
  - *Кластеры* – результат кластеризации, *классы* – результат ручной разметки

# Матрица несоответствий

| | КЛАССЫ | | | |
|---|---|---|---|---|
| КЛАСТЕРЫ | | A | B | C |
| | *a* | 2 | 2 | 0 |
| | *b* | 2 | 2 | 0 |
| | *c* | 0 | 0 | 8 |

- способ примитивный, зато наглядный

# Метрики заимствованные из информационного поиска

| | Релевантные | Нерелевантные |
|---|---|---|
| Найденные | tp | fp |
| Ненайденные | fn | tn |

Полнота (recall):

*R = tp / (tp+fn)*

Точность (presicion):

*P = tp / (tp+fn)*

F-мера:

$$F_\alpha = \frac{(1+\alpha)RP}{\alpha P + R}$$

Accuracy):

*Acc = (tp +tn) / (tp +fp +fn)*

# Применительно к кластеризации

$$F = \sum_i \frac{n_i}{n} \operatorname*{argmax}_j F(i,j)$$

*i – классы, j – кластеры, n – общее число документов, $n_i$ – число документов в классе i*

Т.е. для каждого класса выбираем кластер, который ему больше соответствует (argmax), суммируем меры соответствия (*F*) для всех классов, при этом чем больше класс, тем больше его вес в общей сумме ($n_i$).

F-мера показывает общее качество кластеризации, но не показывает как устроены сами кластеры.

# Чистота

$$Purity = \sum_j \frac{n_j}{n} \operatorname*{argmax}_i P(i,j)$$

*i – классы, j – кластеры, n – общее число документов, $n_j$ – число документов в кластере j, P(i,j) – доля документов из класса i в кластере j .*

Т.е. берем долю доминирующего (argmax) класса в кластере (*P(i,j)*), и суммируем по всем кластерам, при этом чем больше кластер, тем больше его вес в сумме ($n_j$ ).

Чем выше значение чистоты, тем лучше. В идеальном случае P=1.

# Энтропия

$$Entropy = -\frac{1}{\log k} \sum_j \frac{n_j}{n} \sum_i P(i,j) \log P(i,j)$$

*i – классы, j – кластеры, n – общее число документов, $n_j$ – число документов в кластере j, P(i,j) – доля документов из класса i в кластере j , k – число кластеров.*

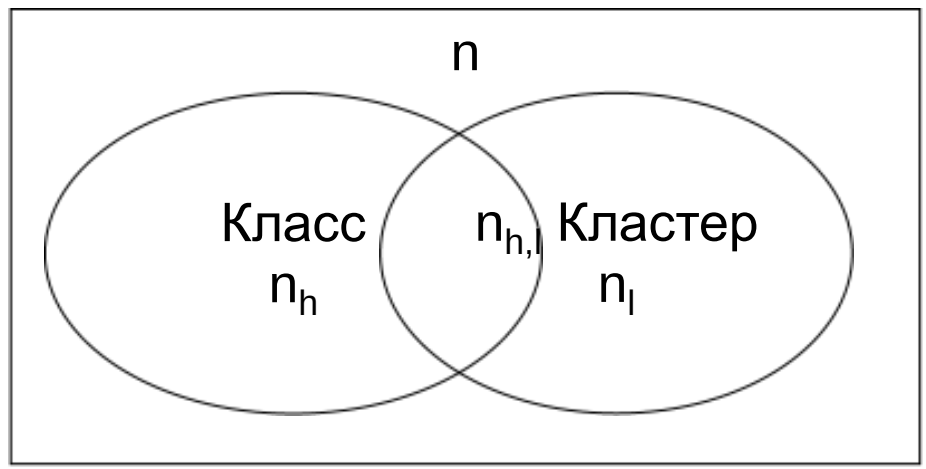
Энтропия – степень «размазанности» класса по кластерам. Чем меньше, тем лучше, в идеале E=0.

# Взаимная информация

- Чистота и энтропия хороши тогда, когда число классов и кластеров совпадает. В других случаях лучше MI (или NMI – нормализованная взаимная информация).

$$NMI = \frac{\sum_{h,l} n_{h,l} \log \frac{n n_{h,l}}{n_h n_l}}{\sqrt{(\sum_h n_h \log \frac{n_h}{n})(\sum_l n_l \log \frac{n_l}{n})}}$$

*n – общее число документов, $n_h$ – число документов в классе h, $n_l$ – число документов в кластере l, $n_{h,l}$ – число документов в пересечении.*



# Стабильность

С помощью взаимной информации можно считать стабильность, т.е. степень пересечения кластеризации при разных прогонах одного и того же алгоритма.

$$\varphi(\mathbf{\Lambda}, \hat{\lambda}) = \frac{1}{r} \sum_i NMI(\hat{\lambda}, \lambda_i)$$

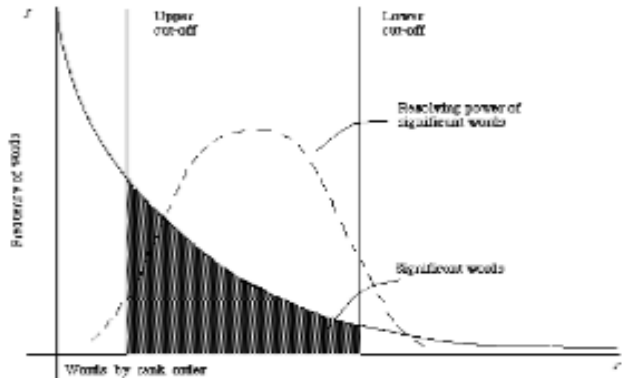
Λ – множество различных кластеризаций, λ – конкретная кластеризация, r – число кластеров.

# 3. Векторная модель

- Коллекция из n документов и m различных терминов представляется в виде матрицы mxn, где каждый документ – вектор в m-мерном пространстве.
- Веса терминов можно считать по разному: частота, бинарная частота (входит – не входит), tf*idf…
- Порядок слов не учитывается (bag of words)
- Матрица очень большая (большое число различных терминов в гетерогенной коллекции).
- В матрице много нулей

# 3.1 Предобработка

- Фильтрация (удаление спецсимволов и пунктуации)
- Токенизация (разбиваем текст на термины – слова или словосочетания)
- Стемминг (приведение слова к основе)
- Удаление стоп-слов
- Сокращение (удаление низкочастотных слов)



# 4. Иерархическая кластеризация

- На начальной стадии каждый документ – сам себе кластер.
- На каждом шаге документы объединяются до построения полного дерева.
- Число кластеров заранее не оговаривается.
- Не подходит для больших объемов данных (подсчет расстояния на каждой стадии).

# «Разделяющая» кластеризация

- Классический пример - kmeans:
- Выбирается k случайных документов, которые считаются центроидами кластеров, все остальные документы распределяются по кластерам по степени близости к центроидам
- На следующих итерациях центроиды пересчитываются и документы перераспределяются
- Косинусная метрика лучше, чем Евклидово расстояние

$$\cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

# Недостатки kmeans

- Результаты могут быть различными в зависимости от инициализации.
- Может останавливаться на субоптимальном локальном минимуме
- Чувствителен к шуму и случайным выбросам
- Вычислительная сложность: $\mathcal{O}(nkl)$

где *n* – число документов, *k* – число кластеров, *l* – число итераций.

# Лирическое отступление: O

- Вычислительная сложность алгоритма O(f(n)), где n – это объем данных – означает, что в худшем случае для выполнения алгоритма понадобится $c_1+c_2*f(n)$ операций, где $c_1$ и $c_2$ константы.

- Примеры:
  - O(n) – линейный алгоритм
  - $O(n^a)$ – полиномиальный алгоритм
  - $O(a^n)$ – экспоненциальный алгоритм (очень плохо)

  - O(n*log(n)) – часто встречается; медленнее линейного, но быстрее полиномиального

# 4.1 Online spherical kmeans (oskmns)

- Документы не обрабатываются все сразу, а **добавляются**
- Новый документ добавляется в тот кластер, к которому он ближе
- Центроид не пересчитывается как среднее всех элементов, а «поправляется»:

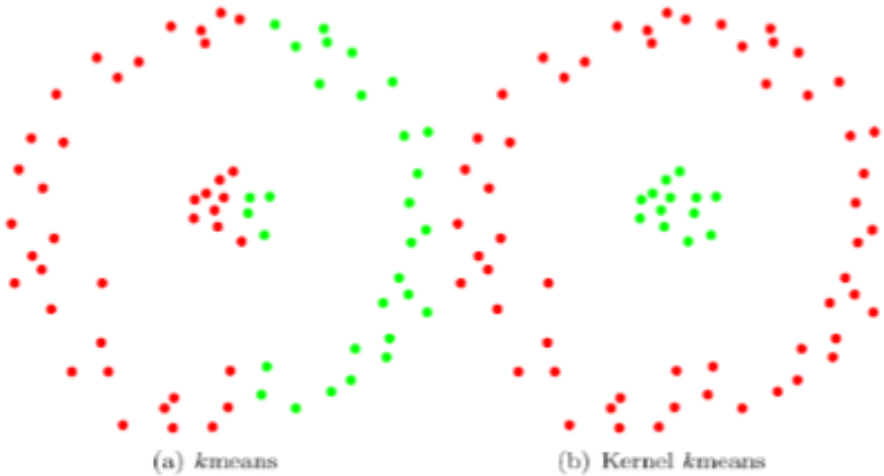
$$\mu^*_{k(x)} = \frac{\mu_{k(d)} + \eta \mathbf{d}}{|\mu_{k(d)} + \eta \mathbf{d}|}$$

*k(x) – кластер, к которому ближе всего документ, μ – центроид, d – вектор документа,η – убывающий коэффициент*

$$\eta_t = \eta_0 \left( \frac{\eta_f}{\eta_0} \right)^{\frac{t}{NM}}$$

*N – число документов, M – число итераций, $\eta_f$ – желаемое конечное значение (0,01)*

# 4.2 Kernel kmeans




kmeans бессилен в тех случаях, кластеры разделяются нелинейно. Kernel kmeans позволяет перейти в пространство более высокой размерности, где кластеры будут разделяться линейно.

- На множестве документов вводится функция, которая отражает степень близости между документами. После этого проводится кластеризация обычным методом kmeans
- Выбор функции очень важен; считается, что для текстовых данных лучше всего косинусная мера близости
- Для конкретных коллекция какая-то другая функция может работать лучше
- Вообще в этом методе пока довольно много сложностей и неопределенности; вычислительно метод очень тяжел
- Существует разновидность, которая порождает нечеткую кластеризацию (fuzzy c-means)

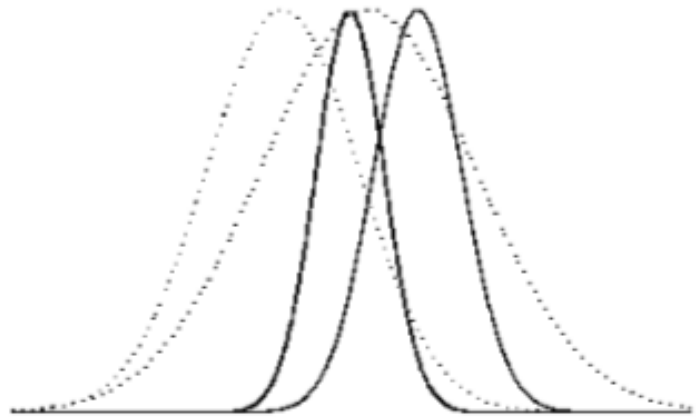
# 5. Генеративные алгоритмы

- Дискриминативные алгоритмы, которые основаны на попарной близости документов, имеют сложность $O(n^2)$ **по определению**.

- Генеративные алгоритмы не требуют такого сравнения, используя итеративные процедуры.

# 5.1 Гауссова модель

- Предполагается, что распределение документов в векторном пространстве – это набор Гауссовых распределений; каждый кластер ассоциирован со средним распределения и матрицей ковариации.



- Ковариация: $$\operatorname{cov}(X, Y) = \sum_{i=1}^{N} \frac{(x_i - \bar{x})(y_i - \bar{y})}{N}.$$
- Если между x и y нет корреляции, то ковариация равна нулю.
- Матрица ковариации: матрица, элементы которой – это попарные ковариации двух векторов.
- Если речь идет об одном и том же наборе векторов (наш случай: одни и те же документы в столбцах и строках), то матрица ковариации – это обобщение дисперсии для многомерной случайной величины.

# Гауссова модель

- Вероятность того, что документ d принадлежит кластеру θ из набора Θ:

$$\mathcal{P}(\mathbf{d}|\theta) = \frac{1}{(2\pi)^{\frac{m}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{(\mathbf{d}-\mu)^T\Sigma^{-1}(\mathbf{d}-\mu)}{2}\right)$$

*P(d| θ) - вероятность того, что документ d принадлежит кластеру θ, m – размерность пространства, μ – сентроид, Σ – матрица ковариации.*

Общая вероятность (правдоподобие того, что данный документ описывается моделью):

$$\mathcal{P}(\mathbf{d}|\Theta) = \sum_{\theta\in\Theta} \mathcal{P}(\theta)\mathcal{P}(\mathbf{d}|\theta)$$

Задача кластеризации: максимизировать это число, максимизировав каждое из слагаемых (т.е. найдя наилучшее среднее и матрицу ковариации для каждого кластера).

# 5.2 Expectation maximization (EM-алгоритм)

- Итеративная процедура для нахождения максимального правдоподобия параметров модели.
- Две стадии:
  - E(xpectation) – вывод скрытых данных из наблюдаемых данных (документы) и текущей модели (кластеры)

$$\mathcal{P}(\theta|\mathbf{d}) = \frac{\mathcal{P}(\theta)\mathcal{P}(\mathbf{d}|\theta)}{\sum_{\theta\in\Theta}\mathcal{P}(\theta)\mathcal{P}(\mathbf{d}|\theta)}$$

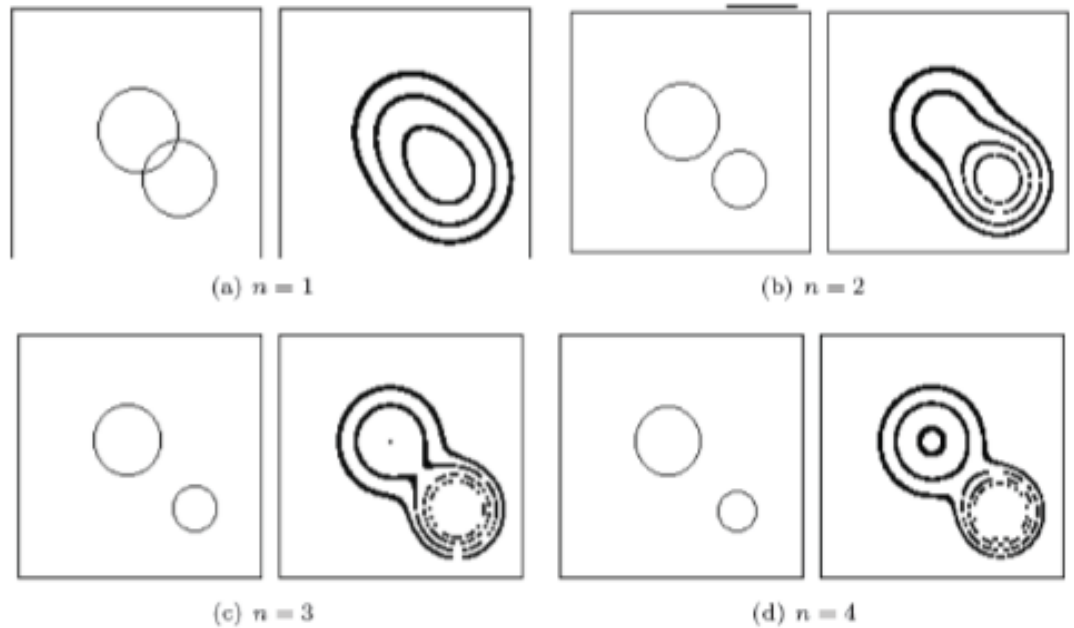
$$\mathcal{P}(\theta)^* = \sum_{d\in\mathcal{D}}\mathcal{P}(\theta|\mathbf{d}).$$

- M(aximization) – максимизация правдоподобия в предположении, что скрытые данные известны

$$\mu = \frac{\sum_{\mathbf{d}\in\mathcal{D}}\mathcal{P}(\theta|\mathbf{d})\mathbf{d}}{\sum_{\mathbf{d}\in\mathcal{D}}\mathcal{P}(\theta|\mathbf{d})}$$

$$\Sigma = \frac{\sum_{\mathbf{d}\in\mathcal{D}}\mathcal{P}(\theta|\mathbf{d})(\mathbf{d}-\mu)(\mathbf{d}-\mu)^T}{\sum_{\mathbf{d}\in\mathcal{D}}\mathcal{P}(\theta|\mathbf{d})}$$

# EM-алгоритм



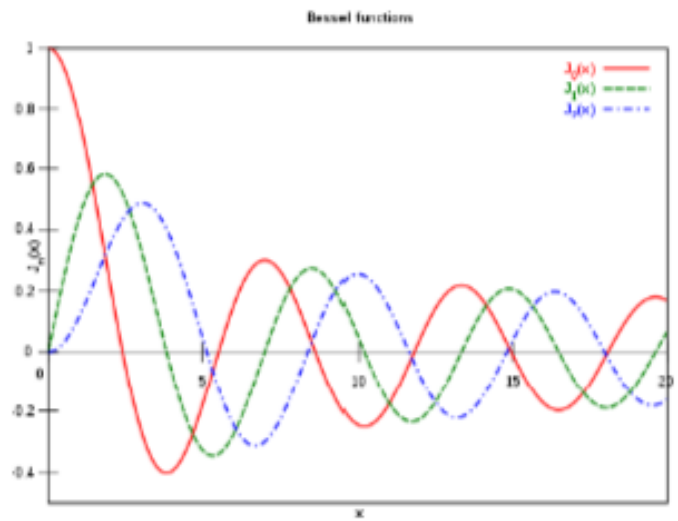

- Большое число свободных параметров может приводить к переобучению.
- Сокращение размерности: выбор дискриминирующих свойств для каждого кластера.
- Сложность: $O(k^2n)$
- Нестабильность, зависимость от инициализации.

# 5.3 Модель фон Мисес-Фишера

- На самом деле, распределение текстов по кластерам гауссианами описывается плохо. Было доказано, что лучше всего подходит vMF-распределение:

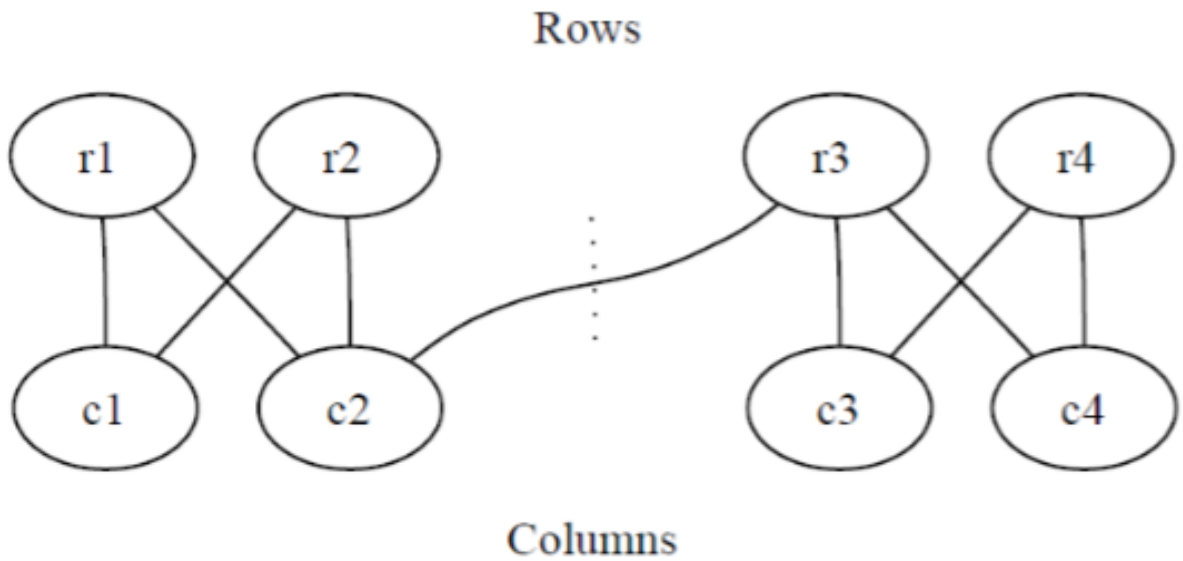
$$\mathcal{P}(\mathbf{d}|\theta) = \frac{1}{Z(\Sigma)} \exp\left(\Sigma \frac{\mathbf{d}^T \mu}{|\mu|}\right)$$

- Z – функция Басселя (фактор нормализации). Затем используют алгоритм, похожий на em. Качество получается лучше, чем spherical k-means.



# 6. Спектральная кластеризация

- Основная гипотеза: термины, которые часто встречаются вместе, описывают близкие понятия. Поэтому важна группировка не только кластеров, но и терминов. Т.е. речь идет о **совместной кластеризации** терминов и документов.
- Матрица термин-документ преобразуется в двудольный граф:



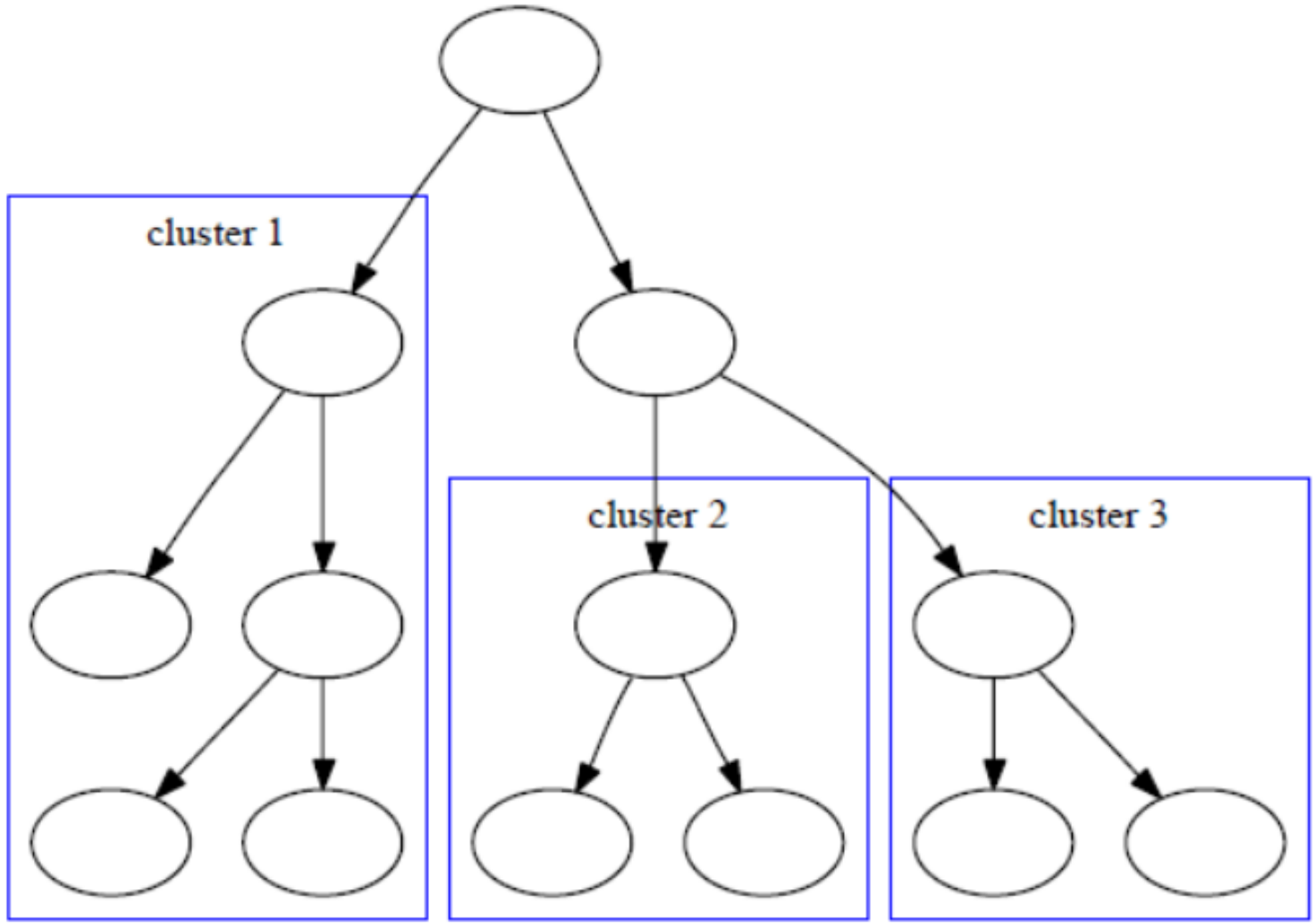

- Тогда задача кластеризации – разбить этот граф на сильно связанные компоненты.
- Почему спектральная: используется сразу несколько функций-критериев разбиения.

# 6.1 Алгоритм divide & merge

- Нахождение оптимального разбиения в графе – NP-полная задача (на практике означает, что алгоритм экспоненциальный). Однако существует аппроксимация.

- Две стадии:
  - Иерархическая кластеризация (существует метод с использованием собственных векторов матрицы, который позволяет избежать неэффективного попарного сравнения)
  - Кластеризация результатов предыдущей стадии с использованием стандартных алгоритмов – kmeans, либо другие алгоритмы, с **неизвестным заранее числом кластеров**

# Алгоритм divide & merge



# 6.2 Нечеткая совместная корреляция

- Кластеризуются сразу и термины, и документы
- Границы между кластерами нечеткие - термин или документ может входить сразу в несколько кластеров (с различными весами)
- Пример: Fuzzy Codok алгоритм

$$u_{ci} = \frac{1}{C} + \frac{1}{2T_u}\left(\sum_{j=1}^{m} v_{cj} d_{ij} - \frac{1}{C}\sum_{j=1}^{m} v_{cj} d_{ij}\right)$$

$$v_{cj} = \frac{1}{K} + \frac{1}{2T_v}\left(\sum_{i=1}^{n} u_{ci} d_{ij} - \frac{1}{K}\sum_{i=1}^{n} u_{ci} d_{ij}\right)$$

*$u_{ci}$ – степень вхождения документа i в кластер c, $v_{cj}$ – степень вхождения термина j в кластер c, $d_{ij}$ – уровень корреляции между документом и термином; m – число терминов, n – число документов, C – число кластеров документов, K – число кластеров терминов.*
*$T_u$, $T_v$ – параметры, их надо подбирать – слабое место алгоритма; оптимальные значения параметров зависят от коллекции.*

# 7. Снижение размерности

- Матрица термин документ A аппроксимируется матрицей меньшего ранга k $A_k$.
- Принятая мера качества такой аппроксимации – норма Фробениуса (чем меньше, тем лучше):

$$|\mathbf{A} - \mathbf{A}_k| = \sqrt{\sum_{a \in \mathbf{A}} \sum_{a_k \in \mathbf{A_k}} (a - a_k)^2}$$

# 7.1 Метод главных компонентов (PCA)

- Главные компоненты – ортогональные (независимые) проекции, которые вместе описывают максимальное разнообразие в данных
- Задача эквивалентна поиску оптимального разбиения в двудольном графе

- Главные компоненты получаются из сингулярного разложения матрицы:

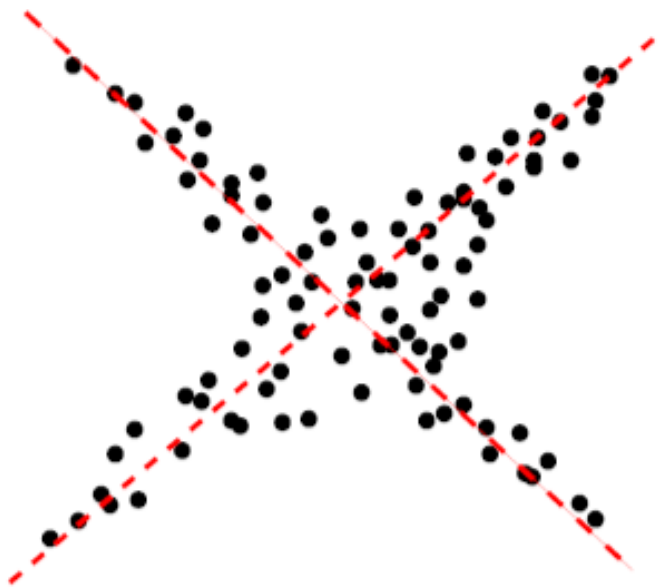
$$A = U\Sigma V^T,$$

U,V – квадратные, Σ – диагональная

- $\Sigma_k$ – диагональная матрица меньшего ранга, в нее входят k наибольших чисел из Σ
- Искомая проекция:

$$A = U\Sigma_k V^T$$

- Чем больше k, тем лучше аппроксимация



# Метод главных компонентов

+ В результате получается оптимальная аппроксимация

+ Различие в расстояниях внутри кластеров и между кластерами становится более резким

- В новом пространстве остаются недискриминирующие свойства – т.е. результаты метода нельзя рассматривать как готовую кластеризацию
- Компоненты должны быть ортогональными – не совсем подходит для текстов, которые могут покрывать несколько тем
- Вычислительно сложный алгоритм, не может использоваться итеративно

# 7.2 Неотрицательная факторизация (NMF)

- Цель: получить аппроксимацию, которая содержит только дискриминирующие факторы
- Исходная матрица аппроксимируется произведением:

$$A \approx UV^T$$

U – базис mxk, V – матрица коэффициентов nxk

- U может интерпретироваться как набор семантических переменных, V - распределение документов по этим темам
- Начальные значения U и V инициализируются случайно, затем итеративно улучшаются (em-алгоритм)
- Мера качества – обычно Евклидово расстояние (чем меньше, тем лучше): $\Theta = \sum_i \sum_j \left( \mathbf{A}_{ij} - \sum_l \mathbf{U}_{il} \mathbf{V}_{jl} \right)$

- Вместо случайно инициализации можно использовать результаты более простого метода кластеризации (skmns)

# 7.3 Мягкая спектральная кластеризация

- Из редуцированного пространства трудно породить нечеткую кластеризацию, потому что усечение матрицы приводит к искажениям

- Выход: независимая кластеризация терминов и документов; на основе кластеризации терминов порождается нечеткая кластеризация документов и vice versa

# Мягкая спектральная кластеризация

- Пространство редуцируется методом главных компонентов
- Проводится кластеризация методом kmeans (или другим)
- Для этих кластеров порождается матрица $P = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix}$
- $P_1$ описывает распределение терминов по кластерам, $P_2$ – документов
- Веса терминов высчитываются с помощью трансформации A $P_2$ – **проекция ценроидов** в исходное пространство
- Аналогичная матрица S порождается из кластеризации исходного пространства $S = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix}$
- $A^T S_1$ используется как функция вхождения для документов, $AP_2$ – для терминов (используются только дискриминирующие термины)
- Хорошее качество для пересекающихся тем, но высокая вычислительная сложность

# 7.4 Lingo

- “description comes first”
- Сокращается размерность пространства
- Базисные вектора полученной редуцированной матрицы воспринимаются как метки кластеров
- Эти метки используются для «поиска» документов (как в информационном поиске)

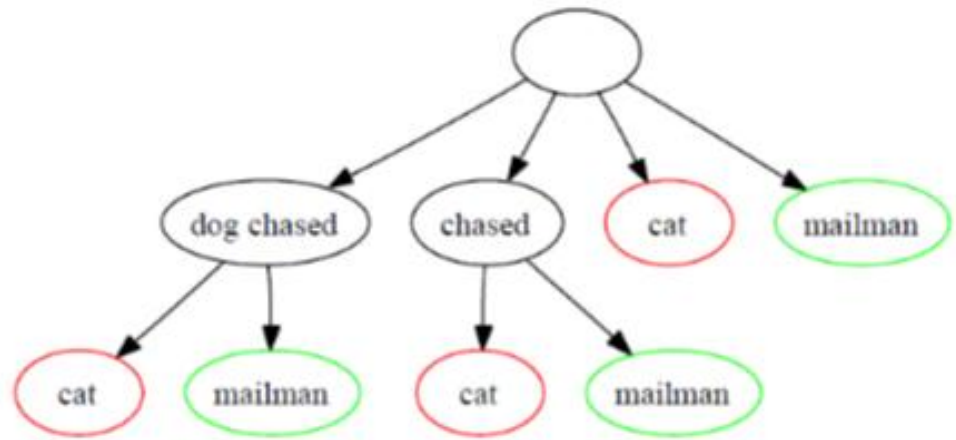
# 8. Модели с учетом порядка слов

- *Маша любит Васю, Вася любит Машу* – векторная модель не учитывает различие, но оно есть

- Гипотеза: учет порядка слов может улучшить качество кластеризации
- Кроме того, он позволит создавать более разумные описания кластеров (не набор слов, а короткие фразы)

# 8.1 Кластеризация на основе суффиксных деревьев

- Суффикс – несколько слов с конца предложения (вплоть до предложения целиком)
- Суффиксное дерево: описывает все общие суффиксы документов




*dog chased cat*,
*dog chased mailman*

- Общие суффиксы используются для выделения базовых кластеров, которые затем объединяются методом связанных компонентов
- Общая сложность алгоритма: O(n log(n))

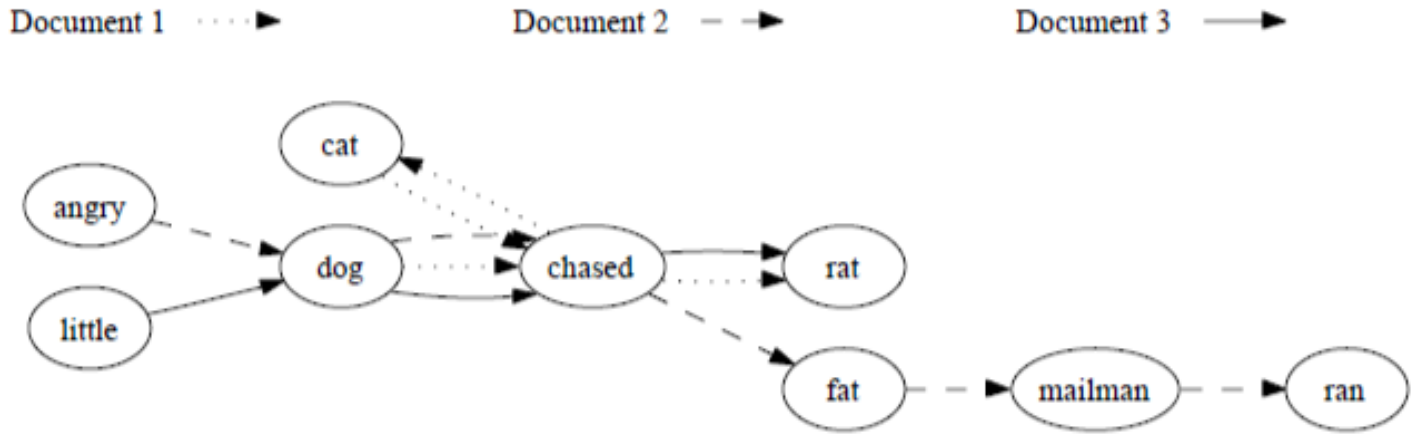
# Кластеризация на основе суффиксных деревьев

- Кластеры включают не все документы (документ может иметь суффикс, которые не пересекается ни с одним другим)
- Не учитывается распределение слов по коллекции (не все слова одинаково полезны)
- Учитываются только совпадающие суффиксы, не совпадающие суффиксы не учитываются
- Проверялось на сниппетах, на длинных текстах и больших коллекциях работает плохо

- Можно совмещать учет порядка слов с обычной косинусной мерой

# 8.2 Граф документа

- Doc1: “*cat chased rat*”, “*dog chased rat*”
- Doc2: *“angry dog chased fat mailman” “mailman ran”*
- Doc3: “*little dog chased ran”*




- Слова хранятся в вершинах, с учетом частоты
- Нет избыточной информации (как в суффиксных деревьях)
- Это не алгоритм кластеризации, а модель документа; мера близости основана на перекрывающихся подграфах
- Лучше работает совместно с косинусной метрикой, но это – двойная стоимость вычислений

# Анализ

| Algorithm | Section | Complexity | Fuzzy | Finds $k$ | Open | Online |
|---|---|---|---|---|---|---|
| Kernel $k$means | 4.2 | - | No | No | No | No |
| RSSC | 7.3 | - | Yes | No | No | No |
| NMF | 7.2 | $\mathcal{O}(k^2 n)$ | No | No | No | No |
| EM vMF | 5.2 | $\mathcal{O}(k^2 n)$ | Yes | No | Yes (Java) | No |
| Fuzzy codok | 6.2 | - | Yes | No | No | No |
| Eigencluster | 6.1 | $\mathcal{O}(kn \log n)$ | No | Yes | No | No |
| Spectral coclustering | 7.1 | $\mathcal{O}(kn^2)$ | No | No | No | No |
| Online spherical $k$means | 4.1 | $\mathcal{O}(kn)$ | No | No | No | Yes |
| CLUTO | 6 | $\mathcal{O}(kn)$ | No | No | Yes (C) | No |
| Lingo | 7.4 | $\mathcal{O}(kn^2)$ | Yes | Yes | Yes (Java) | No |
| STC | 8.1 | $\mathcal{O}(n \log n)$ | Yes | Yes | Yes (Java) | Yes |

- Качество кластеризации определяется по стандартным мерам, при этом итоговая кластеризация не всегда выглядит «естественно»
- Проблема инициализации (итеративные алгоритмы используют случайную инициализацию)
- Проблема описания кластеров (меток)
- Проблема числа кластеров
- Существуют другие методы кластеризации, возможно, они окажутся хороши для текстовых данных
- Возможно другие меры близости, помимо косинусной, окажутся применимы

# Спасибо за внимание!