Статистическая и признаковая модели текста

Ефремова Наталья Эрнестовна
Грацианова Татьяна Юрьевна

Содержание

- Признаковые модели текста
  - виды признаков
  - примеры моделей
  - задание весов признаков
- Признаковая модель в задачах области Text Mining
  - задача классификации и ее приложения
  - задача кластеризации и ее приложения
- Статистические языковые модели
- Применение статистических моделей

Статистическая языковая модель

- Отвечает на вопрос, насколько данная фраза типична/правильна для языка
- Типичность/правильность можно задать с помощью вероятности

Статистическая языковая модель (*Language Model*) – **приписывание** любой фразе/предложению языка вероятности ее появления в тексте

- При создании модели:
  - ✓ вычисляют вероятности (по корпусу)
  - ✓ устраняют ее недостатки – сглаживают
  - ✓ оценивают качество построенной модели

Языковые модели. Разновидности

Вероятность может быть приписана:

- каждой фразе/предложению языка

  Пусть в языке 1000 слов, а длина предложения – не более 10. Число возможных предложений:

  $$1000^1 + \ldots + 1000^{10} =$$
  $$= 1001001001001001001001001001000$$

  Необходимо хранить ≈ 10^{30} (2^{100}) чисел, большая часть из которых = 0, т.к. многие предложения либо неверны, либо раньше не встречались

- лексико-синтаксическим конструкциям
- последовательностям из N слов – N-граммная модель (более компактная)

N-граммная модель

- Рассматривают N-граммы – последовательности из N слов $W = w_1\ w_2\ \ldots\ w_N$. Обычно $N = 1, 2, 3$ или 4
- Предположение: вероятность (P) появления очередного слова в предложении зависит только от предыдущих $N-1$ слов

  $P(W)=P(w_1)*P(w_2|w_1)*P(w_3|w_1\ w_2)*\ldots*P(w_N|w_1\ w_2\ \ldots\ w_{N-1})$
- P всего предложения вычисляется как произведение P входящих в него N-грамм

Вода так прозрачна, что хочется …

Для $N=4$:

*P(вода так прозрачна что хочется) = P(вода)*P(так|вода)* *P(прозрачна|вода так)*P(что|вода так прозрачна)* *P(хочется|так прозрачна что)*

Как вычислить вероятность

- **Теория**: вероятности вычисляются по корпусу с опорой на частоты фраз

$$P(w_N|w_1\ w_2\ \dots\ w_{N-1}) = Fr(w_1\ w_2\ \dots\ w_N) / Fr(w_1\ w_2\ \dots\ w_{N-1})$$

- **Реальность – ограниченность корпуса**: чем больше N, тем большего количества правильных N-грамм в корпусе нет ➔ $P(w_N|w_1\ w_2\ \dots\ w_{N-1})$ не всегда можно верно вычислить

Марковская модель: учитывается только $k < N-1$ предыдущих слов. При $k=2$ получим

$$P(w_1\ w_2\ \dots\ w_N) \approx P(w_1)*P(w_2|w_1)*P(w_3|w_2)*\dots*P(w_N|w_{N-1})$$

Пример: вычисление вероятности предложения

- Для простоты рассмотрим $N=2$
- Вычислим P для предложения *I want to eat*:

$$P(I\ want\ to\ eat) = P(I) * P(want|I) * P(to|want) * P(eat|to)$$

- $P(I) = Fr(I)$, $P(want|I) = Fr(I\ want) / Fr(I)$ и т.д.
- По корпусу Berkeley Restaurant Corpus получено:

$$P(I) = 0{,}25 \qquad P(want|I) = 0{,}32$$
$$P(to|want) = 0{,}65 \qquad P(eat|to) = 0{,}26$$

- Следовательно,

$$P(I\ want\ to\ eat) = 0{,}25*0{,}32*0{,}65*0{,}26 \approx 0{,}014$$

Плюсы и минусы N-граммной модели

+ Возможность построения модели по корпусу
+ Относительная простота использования

− Предположение о независимости вероятности от более длинной истории
The computer which I had just put into the machine room on the fifth floor crashed

− Колоссальные объемы хранимой информации
Если в языке 1000 слов, то получим:
10^6 биграмм, 10^9 триграмм, 10^{12} тетраграмм

− Недостаточность данных для построения модели (ограниченность корпуса и его специфика)
P=0 у неправильной N-граммы – ☺
P=0 у N-граммы, которой нет в корпусе – ☹

Сглаживание

- Недостаточность данных для построения модели ➔ некорректные вероятности

  Пусть в корпусе есть фраза *I want to eat*, но нет фразы *I want to eat British food*

  Получается, что $P(British \mid I\ want\ to\ eat) = 0$ и $P(I\ want\ to\ eat\ British\ food) = 0$

- Для устранения некорректности применяется **сглаживание**: понижают P одних и повышают P других N-грамм
- Один из способов – добавление $\alpha \leq 1$

$$P(w_2 \mid w_1) = \frac{Fr(w_1 w_2) + \alpha}{Fr(w_1) + \alpha V},$$

где V – количество различных слов в корпусе

Сглаживание: пример (1)

- Пусть в корпусе нет фразы *British food*. Тогда:

  $P(British|eat) = 0 \qquad P(food|British) = 0$

  $P(I\ want\ to\ eat\ British\ food) = P(I\ want\ to\ eat)\ *$
  $* P(British|eat) * P(food|British) \approx 0{,}014 * 0 * 0 = 0$

- Пусть размер корпуса 14180 словоупотреблений, 1616 слов

- Пусть по корпусу получено:

  $Fa(I\ want) = 1134$ $\qquad Fa(I) = 3545$
  $Fa(want\ to) = 793$ $\qquad Fa(want) = 1220$
  $Fa(to\ eat) = 942$ $\qquad Fa(to) = 3623$
  $Fa(eat\ British) = 0$ $\qquad Fa(eat) = 3261$
  $Fa(British\ food) = 0$ $\qquad Fa(British) = 0$

Сглаживание: пример (2)

- Возьмем $\alpha = 1$, $V = 1616$. Тогда по формуле:

$$P(British \mid eat) = \frac{Fa(eatBritish)+1}{Fa(eat)+V} = \frac{0+1}{3261+1616} \approx 0,0002$$

$$P(food \mid British) = \frac{0+1}{0+1616} \approx 0,0006$$

- При пересчете $P(I\ want\ to\ eat)$ получим $\approx 0,003$
- Теперь вычислим P для предложения *I want to eat British food*:

$$P(I\ want\ to\ eat\ British\ food) = P(I\ want\ to\ eat) * * P(British|eat) * P(food|British) \approx$$

$$\approx 0,003 * 0,0002 * 0,0006 \approx 0,00000000036$$

Оценка модели: перплексия

- Пусть по текстовой выборке построена N-граммная модель. Как оценить ее качество?
- Предлагается оценить вероятность появления некоторого текста в рамках построенной модели
- Используется перплексия (perplexity) – величина, обратная средней вероятности появления слов в тестируемом тексте

$$PP = P_a(w_1 w_2 \ldots w_V)^{-\frac{1}{V}} = \sqrt[V]{\prod_{i=1}^{V} \frac{1}{P(w_i \mid w_1 \ldots w_{i-1})}},$$

где V – объем тестируемого текста

Перплексия: пример

- Перплексию также можно трактовать как среднее количество слов, которые могут следовать за некоторым фиксированным словом
- Таким образом, чем больше перплексия, тем выше неоднозначность, и следовательно, хуже языковая модель

Пример. Объем корпуса для построения моделей – 38 млн. словоупотреблений, объем V – 1,5 млн. словоупотреблений

	униграммы	биграммы	триграммы
PP	*962*	*170*	*109*

Области применения языковых моделей

- Выявление разного рода ошибок
  - ✓ опечатки: *P(на странице 25) > P(на страннице 25)*
  - ✓ неверный порядок слов: *P(the bed is) > P(the is bed)*
  - ✓ некорректное употребление слов:
    P(my house is small) > P(my home is small)
- Машинный перевод: выбор слова при переводе (*a herd of horses* – *табун* или *стадо лошадей*)
- Распознавание речи: некоторые различные по написанию слова произносятся одинаково. Нужно выбрать в контексте правильное слово
  P(у меня грипп) > P(у меня гриб)
  P(recognize speech) P(wreck a nice beach)
- Генерация текста, например, составление рефератов и аннотаций

Признаковая модель текста

- Для части прикладных задач КЛ (классификации/ кластеризации, извлечения информации и др.) не требуется полного понимания смысла текста, достаточно правильно описать его содержание

  ➔ не нужна **модель** всего **языка**, подойдет **модель** обрабатываемого **текста**

- Модель текста позволяет сравнивать тексты друг с другом и единообразно обрабатывать их коллекции
- Сейчас наиболее распространена и практически значима *признаковая* модель:
  текст – неупорядоченный набор (множество) *информационных признаков* (*features*)

Виды признаков

- **Лингвистические признаки:**
  - ✓ слова: обычно значимые, не служебные; реже – контексты слов и словосочетания
  - ✓ N-граммы (шинглы)
- **Статистические признаки текста**:
  - ✓ количество слов с большой буквы
  - ✓ доля различных частей речи в тексте
  - ✓ доля сложных предложений
  - ✓ средняя длина слова/предложения и т.д.
- **Экстралингвистические признаки**:
  - ✓ тип документа, автор, заголовок
  - ✓ дата публикации, источник информации
  - ✓ гиперссылки и пр.

Модель «мешок слов» (Bag of Words, BOW)

- Признаки – значимые слова текста (*terms*)
- Не учитываются:
  - связи между словами
  - порядок слов в тексте
  - грамматические формы слов
- Если имеется N текстов и каждый текст – набор слов:

$$w_j = (w_{j1}, \ldots, w_{jm}), \quad \text{где}$$

w_{ji} – *вес* i-ого слов в j-ом тексте
m – число слов в текстах

По сути, имеем вектор в m-мерном пространстве

- Вес может просто отображать наличие или отсутствие слова ($w_{ji} = 0$ или 1)

«Мешок слов». Пример

1: ***Карл у Клары украл кораллы***
2: *Клара у Карла украла кларнет*
3: ***Клара у Карла украла кораллы***
4: *Простота – хуже воровства*

Слова-признаки:
[1]*Карл*, [2]*Клара*, [3]*украсть*, [4]*коралл*,
[5]*кларнет*, [6]*простота*, [7]*хуже*, [8]*воровство*

Вес: присутствует признак в документе или нет

$w_1=(1,1,1,1,0,0,0,0)$ $w_2=(1,1,1,0,1,0,0,0)$

$w_3=(1,1,1,1,0,0,0,0)$ $w_4=(0,0,0,0,0,1,1,1)$

N-граммная модель

- Берем последовательности из N слов
- Рассматриваем все слова текста, не только значимые

This example not bad. It helps us.

Пусть $N=2$

this example
example not
not bad
bad it
it help
help us

Задание весов признаков

В общем случае вес i-ого признака в j-ом тексте задается следующим образом:

$$w_{ji} = l_{ji} g_i n_j, \text{ где}$$

- l_{ji} – локальный вес признака в тексте
- g_i – глобальный вес признака во всей коллекции текстов
- n_j – нормирующий множитель для текста

Основой для задания весов обычно служит f_{ji} – частота встречаемости i-ого признака в j-ом тексте

Задание весов признаков. Варианты

Пусть $l_{ij} = f_{ji}$, $g_i = 1$

- Простая частота признака:

$$n_j = 1 \quad \rightarrow \quad w_{ji} = f_{ji}$$

- Нормализованная частота:

$$n_j = \frac{1}{\sqrt{\sum_{i=1}^{m} (l_{ji} g_i)^2}} \quad \rightarrow \quad w_{ji} = \frac{f_{ji}}{\sqrt{\sum_{i=1}^{m} (f_{ji})^2}}$$

Задание весов признаков. tf-idf

- tf-idf опирается на предположение:
  при обработке коллекции текстов частотные признаки менее информативны, чем редкие
- Веса частотных признаков должны быть ниже, чем веса редких

  $l_{ij} = f_{ji} = tf_{ji}$ (*term frequency*), $n_j = 1$

  $g_i = \log(\frac{N}{N_i}) = idf_i$ (*inverse document frequency*), где

  N_i – число текстов, в которых есть i-ый признак

$$w_{ji} = f_{ji} \log(\frac{N}{N_i}) = tf_{ji} idf_i$$

tf-idf. Пример

1: *Мама мыла губкой Машу*
2: *Мама мыла, мыла раму*
3: *В магазине мама купила губку*

	мама	мыть	губка	Маша	рама	магазин	купить
df_i	3	2	2	1	1	1	1
idf_i	0	0,18	0,18	0,47	0,47	0,47	0,47

Документ 1			Документ 2			Документ 3		
слово	tf_{ji}	TF-IDF	слово	tf_{ji}	TF-IDF	слово	tf_{ji}	TF-IDF
Маша	1	0,47	рама	1	0,47	магазин	1	0,47
губка	1	0,18	мыть	2	0,36	купить	1	0,47
мыть	1	0,18	мама	1	0	губка	1	0,18
мама	1	0				мама	1	0

Задание весов признаков. Странность (Weirdness)

- Термины типичны для научной, но не художественной прозы; оценочные слова типичны для отзывов, но не для новостей и пр.
- *Странность* призвана зафиксировать более высокую долю определенных слов в целевом корпусе по сравнению с контрастным

$$\frac{f_{iS} / N_S}{f_{iG} / N_G}$$, где

f_{iS} – частота употребления i слова в корпусе S
N_S – общее количество слов в корпусе S
f_{iG} – частота i слова в контрастном корпусе G
N_G – общее количество слов в корпусе G

Достоинства и недостатки модели

+ Простота модели
+ Для векторов удобно вычислять меру близости (например, косинусную меру)
– В векторах много нулевых весов, т.к. в одном тексте редко «встречаются» сразу все признаки
– Много малоинформативных признаков – тех, которые «встречаются» только в одном-двух или сразу во всех текстах ➔ необходимо уменьшать количество признаков

Признаковая модель используется в задачах Text-mining: классификация и кластеризация документов, извлечение информации и пр.

Уменьшение количества признаков

Уменьшение приводит к:

- упрощению процедур обработки текстов, повышению их надежности и устойчивости
- обозримости пространства признаков, возможности его анализа

Используемые методы:

- Методы селекции: из исходного множества признаков отбираются наиболее полезные
- Методы трансформации: строятся новые признаки, являющиеся комбинацией исходных, например, заменяющие группы синонимичных слов

Классификация и кластеризация. Основные понятия

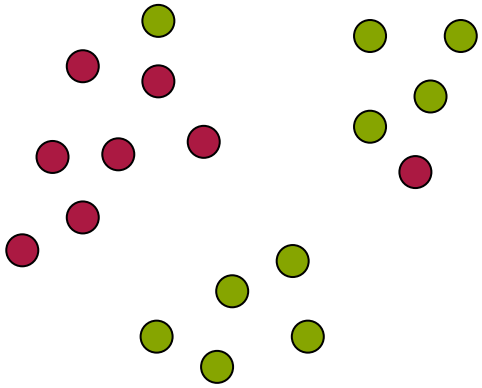
Классификация (рубрицирование) – отнесение документа к заранее известным классам (рубрикам)

- часто документы классифицируют по их содержанию
- классы: с заданными характеристиками, иерархическая система классов

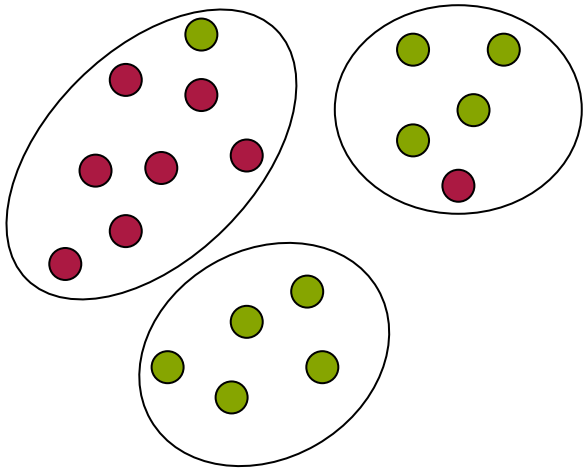
Кластеризация – разбиение заданного множества документов на подмножества (кластеры)

- документы в кластерах похожи по смыслу/стилю/теме/структуре/…
- характеристики, количество, структура кластеров заранее не заданы

Отличие классификации от кластеризации



Классификация: классы изначально предопределены



Кластеризация: кластеры не предопределены, ищем однородные группы:

- объекты внутри группы «похожи»
- объекты разных групп как можно более отличны

Автоматическая классификация

- Имеется множество классов/рубрик/категорий
  $C = \{c_1, \ldots, c_{|C|}\}$
- Имеется множество документов
  $D = \{d_1, \ldots, d_{|D|}\}$
- Есть неизвестная целевая функция
  $\Phi: D \times C \rightarrow \{0,1\}$
- Для разбиения C на D необходимо построить классификатор Φ', максимально близкий к Φ
  $\Phi': D \times C \rightarrow \{0,1\}$ – точный ответ
  $\Phi': D \times C \rightarrow [0,1]$ – степень подобия

Этапы обработки текстов

1. **Подготовка входных данных**
   построение моделей (образов) документов:
   каждый текст – вектор в пространстве признаков
2. **Построение классификатора**
   - ✓ «наивный» байесовский классификатор
   - ✓ деревья принятия решений
   - ✓ алгоритм k-ближайших соседей
   - ✓ алгоритм опорных векторов и др.
3. **Классификация**
4. **Оценка качества классификации**
   - ✓ полнота, точность, F-мера
   - ✓ сравнение классификаторов на тестовых наборах

Построение классификатора

- Имеется множество D' вручную размеченных документов, для которых значения Φ известны
- При этом:
  - либо документы отклассифицированы
  - либо для каждой рубрики есть множество положительных и отрицательных примеров
- Множество документов D' делят на две части:
  - обучающая (для построения Φ')
  - проверочная (для проверки его работы)

Предположение:
обучающие и новые данные похожи

Оценка качества классификации

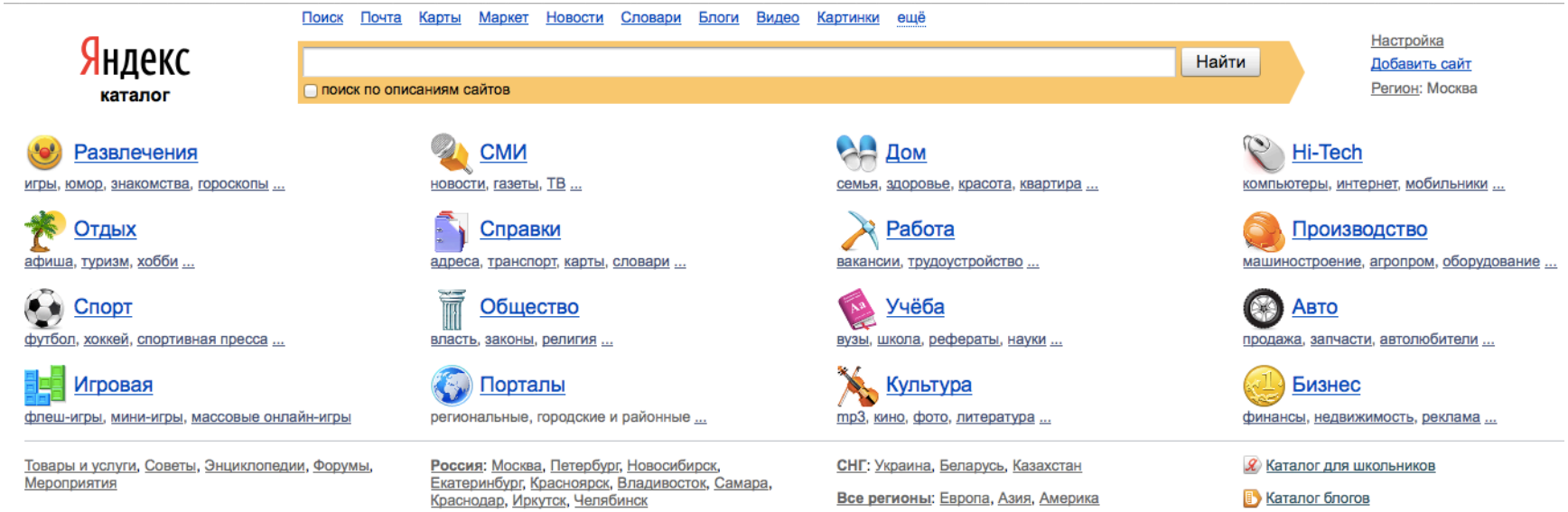
- Вычисляются полнота, точность, F-мера, ошибка классификатора (доля неправильных решений) на тестовом подмножестве
- Используется
  - макроусреднение: составляются отдельные таблицы принятия решений для каждого класса, вычисляются меры, берется среднее по всем классам
  - микроусреднение: составляется единая таблица для всех классов, затем по этой таблице вычисляют меры

Классификация: прикладные задачи

- Упорядочивание набора документов, в частности, распознавание авторства текстов, плагиата и др.
- Навигация по набору документов, в том числе, составление интернет-каталогов
- Ограничение области поиска (в поисковых системах)
- Фильтрация потока документов, например, спама
- Персонализированный/тематический подбор информации: контекстная реклама, новости об определенном событии и т.п.

Классификация: пример

- Тематическое дерево Яндекс: на первом уровне – 16 тем, в каждой – иерархический рубрикатор, число уровней в глубину ≤ 6
- Внутри рубрик – сайты по индексу цитируемости (число ссылок с других сайтов); можно задать регион и тип информационной потребности (советы, форумы, …)



ОБУЧЕНИЕ С УЧИТЕЛЕМ

- В текстовой выборке размечаем различные значения одного слова
- Формируем признаковое представление текста
  - ✓ признаки: соседние слова, коллокации, части речи, синтаксические отношения и др.
- С помощью алгоритмов машинного обучения выявляем значимые признаки
  - ✓ методы классификации по заданному набору классов: SVM, kNN, наивный байесовский классификатор, деревья решений и пр.
- С помощью полученных признаков снимаем неоднозначность в неразмеченных текстах

НАИВНЫЙ БАЙЕСОВСКИЙ КЛАССИФИКАТОР

- Оцениваем вероятность $P(S|F_1, F_2, \ldots, F_n)$ того, что значение S рассматриваемого слова зависит от признаков $F_1, F_2, \ldots, F_n$

$$P(S \mid F_1, F_2, \ldots, F_n) = \frac{P(F_1, F_2, \ldots, F_n \mid S) * P(S)}{P(F_1, F_2, \ldots, F_n)}$$

$P(S)$ – вероятность встретить значение S

$P(F_1, F_2, \ldots, F_n \mid S)$ – вероятности встретить данные признаки для рассматриваемого значения S

- Считаем, что признаки не зависят друг от друга

- Знаменатель не влияет на результат.

$$P(F_1, F_2, \ldots, F_n \mid S) = P(F_1 \mid S) * P(F_2 \mid S) * \ldots * P(F_n \mid S)$$

БАЙЕСОВСКИЙ КЛАССИФИКАТОР. ПРИМЕР

$$sence = \underset{s}{\operatorname{argmax}} \frac{P(F_1 \mid S) * P(F_2 \mid S) * \ldots * P(F_n \mid S) * P(S)}{P(F_1, F_2, \ldots, F_n)}$$

Пусть есть 2000 примеров для слова *bank*:
1500 для $bank_1$ (финансы) и 500 для $bank_2$ (река)

$P(S_1)=1500/2000=0,75$ $P(S_2)=500/2000=0,25$

Признак – слово $F_1=credit$ – встретилось:
200 раз с $bank_1$ и 4 раза с $bank_2$

$P(F_1)=204/2000=0,102$

$P(F_1|S_1)=200/1500=0,13(3)$ $P(F_1|S_2)=4/500=0,008$

$$sence = \underset{s}{\operatorname{argmax}} \begin{bmatrix} \frac{P(F_1 \mid S_1) * P(S_1)}{P(F_1)} \\ \frac{P(F_1 \mid S_2) * P(S_2)}{P(F_1)} \end{bmatrix} = \underset{c}{\operatorname{argmax}} \begin{bmatrix} \frac{0,133 * 0,75}{0,102} \approx 0,978 \\ \frac{0,008 * 0,25}{0,102} \approx 0,02 \end{bmatrix} = S_1$$

ДЕРЕВЬЯ РЕШЕНИЙ

Деревья решений (*Decision Tree*) учитывают встречаемость разных слов в контексте слова W

- Слова, часто встречающиеся с одним из значений слова W, и не встречающиеся с другим, получают высокий ранг $DT\text{-}score$
- Затем слова сортируются по рангу
- Для снятия неоднозначности – ответы на вопросы вида, входит ли признак F_i в контекст W
- Первое совпадение проставляет значение

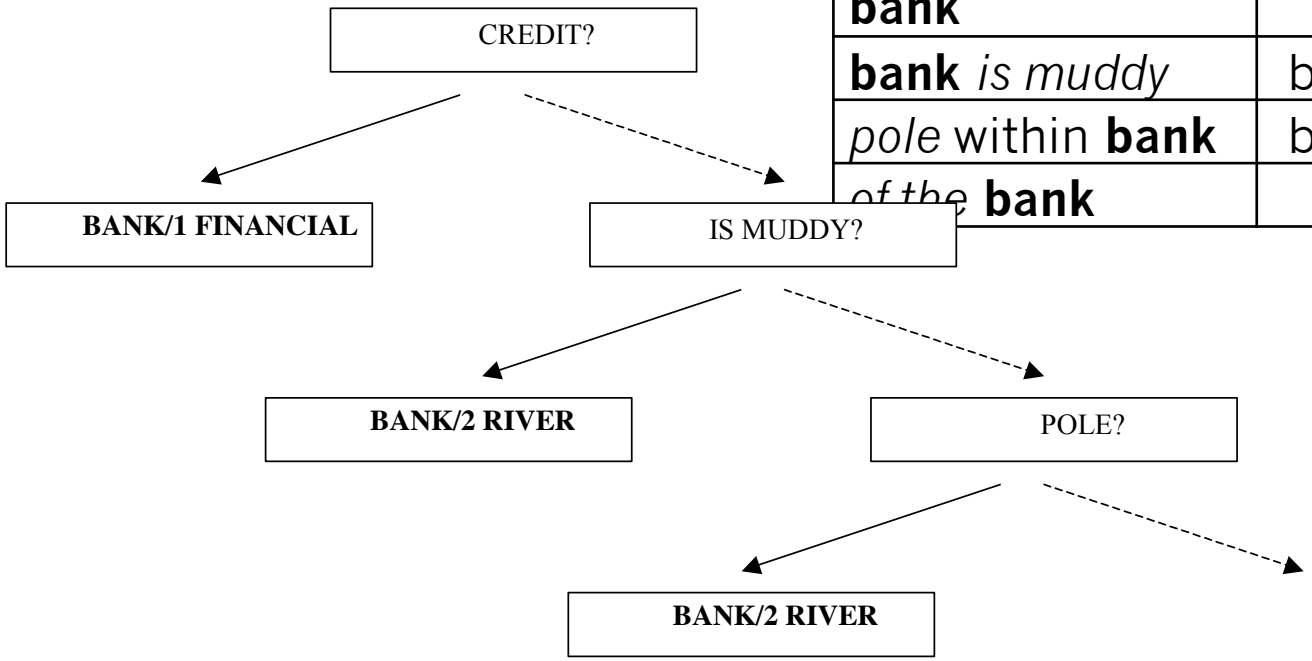
Для предыдущего примера, где F_1=*credit*:

$$DT - score(F_1) = \left| \log \frac{P(S_1 \mid F_1)}{P(S_2 \mid F_1)} \right| = \left| \log \frac{0,978}{0,02} \right| = 3,89$$

ДЕРЕВЬЯ РЕШЕНИЙ. ПРИМЕР

Пусть вычислены ранги для разных слов => можно построить дерево решений:

Feature	Sence	*DL-score*
credit within **bank**	bank/1	3,89
bank *is muddy*	bank/2	2,20
pole within **bank**	bank/2	1,09
of the **bank**	N/A	0,00



Автоматическая кластеризация

- Имеется множество документов $D = \{d_1, \ldots, d_{|D|}\}$
- Необходимо их разбить на подмножества – **кластеры** похожих документов $C = \{c_1, \ldots, c_{|C|}\}$
- Используется понятие схожести документов
  - в идеале: семантическое сходство
  - на практике: документы есть векторы в пространстве признаков, вычисляем расстояние между ними
- Алгоритм должен самостоятельно принимать решение о количестве и составе кластеров
  - **плоские алгоритмы** создают неструктурированное множество кластеров
  - **иерархические алгоритмы** создают структурированное множество кластеров

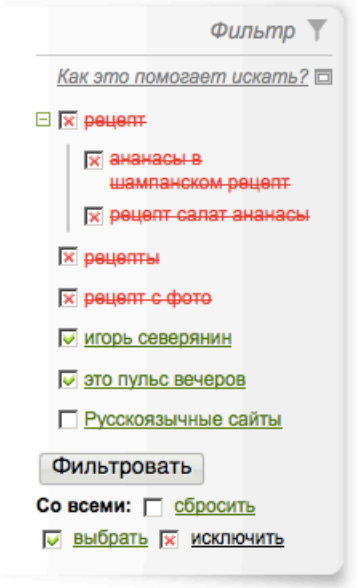
Оценка качества кластеризации

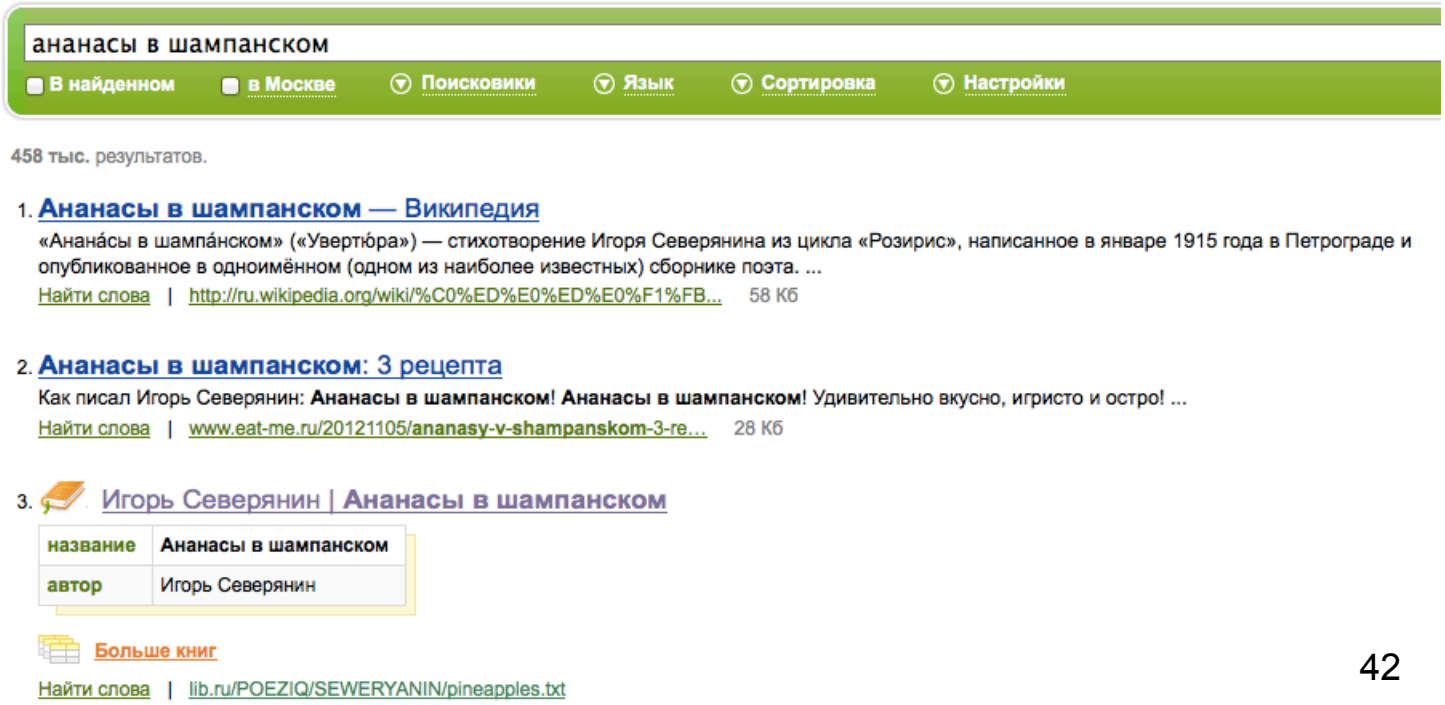
Вычисляются меры двух видов

- Внешние меры: сравнение созданного разбиения с «эталонным»
  анализируется сходство предсказаний экспертов и системы по поводу отнесения каждой **пары** документов **одному или разным** кластерам
- Внутренние меры: анализ внутренних свойств
  - компактность – члены одного кластера должны быть близки друг другу
  - отделимость – кластеры должны далеко отстоять друг от друга

Кластеризация: прикладные задачи

- Упорядочивание набора документов
- Навигация по набору документов
- Информационный поиск:
  - ✓ улучшение результатов поиска
  - ✓ «интеллектуальная» группировка результатов





Выбор алгоритмов классификации и кластеризации

Сложности классификации:

- обучающие документы могут некорректно отражать реальные данные (ошибки, устаревание)
- класс целевой функции может быть не известен

Сложности кластеризации:

- количество кластеров заранее неизвестно
- не существует однозначно наилучшего критерия качества кластеризации
- нет общепризнанных тестовых данных
- результат существенно зависит от того, как определяется схожесть документов

Одного оптимального алгоритма не существует. Главное основание для выбора алгоритма – знание о его теоретических характеристиках и оценка пригодности для решения поставленной задачи

ДИСТРИБУТИВНАЯ СЕМАНТИКА

- Основана на изучении окружения – *дистрибуции*, *распределения* – единиц в тексте
- Лексические единицы (слова, словосочетания) имеют сходство в значении, если они употребляются в схожих контекстах: *кофе, чай, сок,..*
- Метод дистрибутивного анализа (сходство в значении по контексту слова) может выявить:
  - Синонимы
  - Антонимы
  - Слова одного семантического класса

  Все это – слова с общими *семами* (элементами смысла)
- Метод дистрибутивной семантики применяется в КЛ для построения *тезаурусов*:
  - По коллекции текстов строятся классы близких по семантике слов (кластеризация слов)

ТЕХНОЛОГИЯ *Word2Vec*

Google, Mikolov

- Статистическая обработка больших текстовых массивов, машинное обучение, кластеризация
- Подсчет статистики совместной встречаемости слов, с учетом и без учета близости слов в контексте (окружении слова)
- Вход – корпус/коллекция текстов
  Выход – набор (пространство) векторов чисел
  - Размерность пространства – обычно несколько сотен
  - Каждому уникальному слову – свой вектор
  - Смысл имеют только расстояния между векторами, а не сами вектора
- Векторное представление слов: векторы слов, имеющие общие контексты в корпусе, близки в построенном пространстве (косинусная мера)

Word2Vec : ПРИМЕРЫ

Упрощенно: близость контекстов – близость слов

Примеры работы: вывод по мере близости

Enter word or sentence: ***avito***

Word Cosine distance:

— awito 0.693721
авито 0.675299
fvito 0.661414
авита 0.659454
irr 0.642429
овито 0.606189
avimo 0.598056

Enter word or sentence: mail

— rambler 0.777771
meil 0.765292
inbox 0.745602
maill 0.741604
yandex 0.696301
maii 0.675455
myrambler 0.674704
zmail 0.657099
mefr 0.655842
jandex 0.655119
gmail 0.652458
в к mail 0.639919

Word2Vec : АРХИТЕКТУРНЫЕ МОДЕЛИ

Обучение: нейронная двухслойная сеть прямого распространения + техника снижения размерности

Две нейросетевые модели (архитектуры), алгоритмы:
Параметр – контекстное окно (=3-10 слов, по умолчанию 5)

- CBOW (*Continuous Bag-of-Words*)
  - Предсказывает слова при заданном контексте
  - Контекст – «мешок слов», например: 4 ближайших слова: 2 предыдущих, 2 последующих
  - Работает быстрее + лучше для больших корпусов (миллионы и миллиарды слов), т.к. частота слов выше
- SkipGrams (*Continuous Skip-n-Grams*)
  - Предсказывает контекст при заданном слове
  - Контекст – *n-граммы*, с учетом пропусков слов
  - Медленнее, но лучше работает для редких слов, не слишком больших корпусов (<100 миллионов)

SkipGrams и CBOW : *кофе*

Взаимозаменяемые слова:

— *коффе 0.734483*
чая 0.690234
чай 0.688656
капучино 0.666638
кофн 0.636362
какао 0.619801
эспрессо 0.599390
кофя 0.595211
цикорий 0.594247
кофэ 0.593993
копучино 0.587324
шоколад 0.585655
капучинно 0.580286
кардамоном 0.566781
латте 0.563224

Характеризующие слова:

— *зернах 0.757635*
растворимый 0.709936
чая 0.709579
коффе 0.704036
mellanrost 0.694822
сублемированный 0.694553
молотый 0.690066
кофейные 0.680409
чай 0.679867
декофеинизирована 0.67856
капучино 0.677856
monoarabica 0.676757

Word2Vec : ПРИМЕНЕНИЕ

- Исправление опечаток и неверной транслитерации слов, например, для *пщщпду*

  — *пщщщпду - 0.723,… гугл - 0.649, поопду - 0.647…*
- Построение классов семантически близких слов для разных приложений КЛ
- Определение аналогии между словами (семантических отношений), например: *Найти такое слово, которое относится к Германии также, как Париж относится ко Франции*

  — *Мюнхен, Берлин, Дюссельдорф, Гамбург, …*
- Расширение запросов к поисковой системе (запросы к поисковику сильно отличаются от других текстов, учет статистики реальных ошибок)
  - генерация контекстно-близких слов
  - оценка важности слов в запросе

Спасибо за внимание!