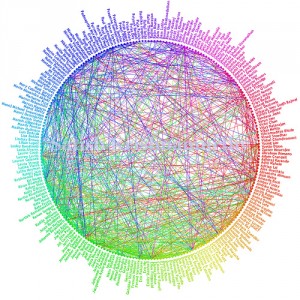
Латентно-семантический анализ

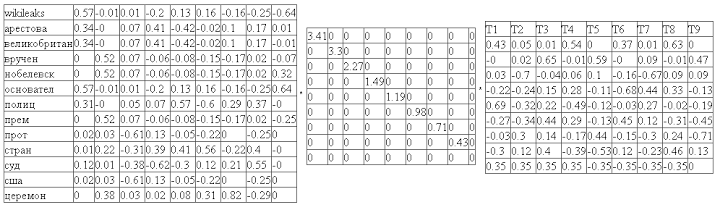
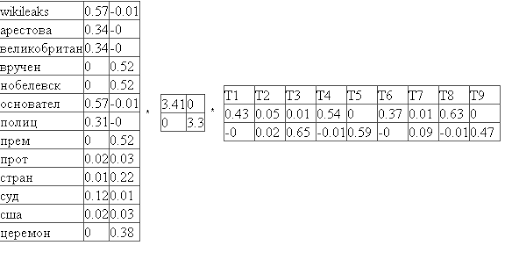
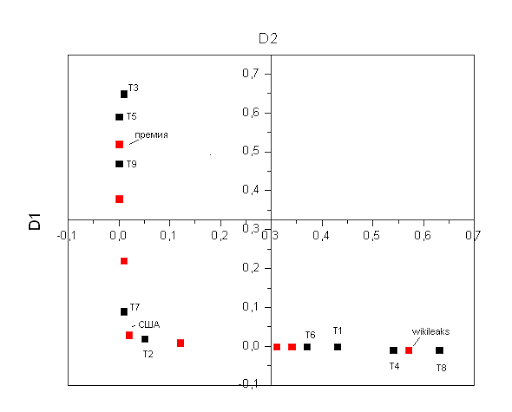
* [Алгоритмы](https://habr.com/hub/algorithms/)
* [Из песочницы](https://habr.com/sandbox/)

Как находить тексты похожие по смыслу? Какие есть алгоритмы для поиска текстов одной тематики? – Вопросы регулярно возникающие на различных программистских форумах. Сегодня я расскажу об одном из подходов, которым активно пользуются поисковые гиганты и который звучит чем-то вроде мантры для SEO aka поисковых оптимизаторов. Этот подход называет [латентно-семантический анализ](http://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D1%82%D0%B5%D0%BD%D1%82%D0%BD%D0%BE-%D1%81%D0%B5%D0%BC%D0%B0%D0%BD%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7)(LSA), он же латентно-семантическое индексирование (LSI)  
  
   
  
  
  
Предположим, перед вами стоит задача написать алгоритм, который сможет отличать новости о звездах эстрады от новостей по экономике. Первое, что приходит в голову, это выбрать слова которые встречаются исключительно в статьях каждого вида и использовать их для классификации. Очевидная проблема такого подхода: как перечислить все возможные слова и что делать в случае когда в статье есть слова из нескольких классов. Дополнительную сложность представляют [омонимы](http://ru.wikipedia.org/wiki/%D0%9E%D0%BC%D0%BE%D0%BD%D0%B8%D0%BC). Т.е. слова имеющие множество значений. Например, слово «банки» в одном контексте может означать стеклянные сосуды а в другом контексте это могут быть финансовые институты.   
  
Латентно-семантический анализ отображает документы и отдельные слова в так называемое «семантическое пространство», в котором и производятся все дальнейшие сравнения. При этом делаются следующие предположения:  
  
1) Документы это просто набор слов. Порядок слов в документах игнорируется. Важно только то, сколько раз то или иное слово встречается в документе.  
2) Семантическое значение документа определяется набором слов, которые как правило идут вместе. Например, в биржевых сводках, часто встречаются слова: «фонд», «акция», «доллар»  
3) Каждое слово имеет единственное значение. Это, безусловно, сильное упрощение, но именно оно делает проблему разрешимой.

Пример

Для примера я выбрал несколько заголовков с различных новостей. Они выбраны не совсем случайно, дело в том, что для случайной выборки потребовался бы очень большой объем данных, что сильно затруднило бы дальнейшее изложение. Итак, было выбрано несколько заголовков.  
  
Первым делом из этих заголовков были исключены, так называемые, стоп-символы. Это слова которые встречаются в каждом тексте и не несут в себе смысловой нагрузки, это, прежде всего, все союзы, частицы, предлоги и множество других слов. Полный список использованных стоп-символов можно посмотреть в моей предыдущей [статье о стоп-симолах](http://www.algorithmist.ru/2010/12/stop-symbols-in-russian.html)  
  
Далее была произведена операция стемминга. Она не является обязательной, некоторые источники утверждают, что хорошие результаты получаются и без нее. И действительно, если набор текстов достаточно большой, то этот шаг можно опустить. Если тексты на английском языке, то этот шаг тоже можно проигнорировать, в силу того, что количество вариаций той или иной словоформы в английском языке существенно меньше чем в русском. В нашем же случае, пропускать этот шаг не стоит т.к. это приведет к существенной деградации результатов. Для стемминга я пользовался [алгоритмом Портера](http://www.algorithmist.ru/2010/12/porter-stemmer-russian.html).  
  
Дальше были исключены слова встречающиеся в единственном экземпляре. Это тоже необязательный шаг, он не влияет на конечный результат, но сильно упрощает математические вычисления. В итоге у нас остались, так называемые, индексируемые слова, они выделены жирным шрифтом:  
  
1. Британская **полиц**ия знает о местонахождении **основател**я **WikiLeaks**  
2. В **суд**е **США**начинается процесс **прот**ив россиянина, рассылавшего спам  
3. **Церемон**ию **вручен**ия **Нобелевск**ой **прем**ии мира бойкотируют 19 **стран**  
4. В **Великобритан**ии **арестова**н **основател**ь сайта **Wikileaks** Джулиан Ассандж  
5. Украина игнорирует **церемон**ию **вручен**ия **Нобелевск**ой **прем**ии  
6. Шведский **суд** отказался рассматривать апелляцию **основател**я **Wikileaks**  
7. НАТО и **США** разработали планы обороны **стран** Балтии **прот**ив России  
8. **Полиц**ия **Великобритан**ии нашла **основател**я **WikiLeaks**, но, не **арестова**ла  
9.В Стокгольме и Осло сегодня состоится **вручен**ие **Нобелевск**их **прем**ий

Латентно семантический анализ

На первом шаге требуется составить частотную матрицу индексируемых слов. В этой матрице строки соответствуют индексированным словам, а столбцы — документам. В каждой ячейке матрицы указано какое количество раз слово встречается в соответствующем документе.   
  
  
  
Следующим шагом мы проводим сингулярное разложение полученной матрицы. [Сингулярное разложение](http://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D0%B3%D1%83%D0%BB%D1%8F%D1%80%D0%BD%D0%BE%D0%B5_%D1%80%D0%B0%D0%B7%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5) это математическая операция раскладывающая матрицу на три составляющих. Т.е. исходную матрицу M мы представляем в виде:   
  
M = U\*W\*Vt  
  
где U и Vt – ортогональные матрицы, а W – диагональная матрица. Причем диагональные элементы матрицы W упорядочены в порядке убывания. Диагональные элементы матрицы W называются сингулярными числами.  
  
  
  
Прелесть сингулярного разложения состоит в том, что оно выделяет ключевые составляющие матрицы, позволяя игнорировать шумы. Согласно простым правилам произведения матриц, видно, что столбцы и строки соответствующие меньшим сингулярным значениям дают наименьший вклад в итоговое произведение. Например, мы можем отбросить последние столбцы матрицы U и последние строки матрицы V^t, оставив только первые 2. Важно, что при этом гарантируется, оптимальность полученного произведения. Разложение такого вида называют двумерным сингулярным разложением:  
  
  
  
Давайте теперь отметим на графике точки соответствующие отдельным текстам и словам, получится такая занятная картинка:  
  
  
  
Из данного графика видно, что статьи образуют три независимые группы, первая группа статей располагается рядом со словом «wikileaks», и действительно, если мы посмотрим названия этих статей становится понятно, что они имеют отношение к wikileaks. Другая группа статей образуется вокруг слова «премия», и действительно в них идет обсуждение нобелевской премии.  
  
На практике, конечно, количество групп будет намного больше, пространство будет не двумерным а многомерным, но сама идея остается той же. Мы можем определять местоположения слов и статей в нашем пространстве и использовать эту информацию для, например, определения тематики статьи. 

Улучшения алгоритма

Легко заметить что подавляющее число ячеек частотной матрицы индексируемых слов, созданной на первом шаге, содержат нули. Матрица сильно разрежена и это свойство может быть использовано для улучшения производительности и потребления памяти при создании более сложной реализации.  
  
В нашем случае тексты были примерно одной и той же длины, в реальных ситуациях частотную матрицу следует нормализовать. Стандартный способ нормализации матрицы [TF-IDF](http://ru.wikipedia.org/wiki/TF-IDF)   
  
Мы использовали двухмерную декомпозицию SVD-2, в реальных примерах, размерность может составлять несколько сотен и больше. Выбор размерности определяется конкретной задачей, но общее правило таково: чем меньше размерность тем меньше семантических групп вы сможете обнаружить, чем больше размерность, тем большее влияние шумов.

Замечания

Для написания статьи использовалась [Java-библиотека для работы с матрицами Jama](http://math.nist.gov/javanumerics/jama/). Кроме того, функция SVD реализована в известных математических пакетах вроде Mathcad, существуют библиотеки для Python и C++.