Санкт-Петербургский государственный университет

Кафедра технологии программирования

Огурцова Анастасия Сергеевна

Выпускная квалификационная работа бакалавра

Автоматическое составление обзора важнейших событий на основе анализа русскоязычных новостных лент

Направление 010400

Прикладная математика и информатика

Научный руководитель,

старший преподаватель

Малинина М. А.

Санкт-Петербург

2017

Содержание

Введение

За последние десятилетие произошел стремительный рост количества электронных новостных ресурсов. С каждым днём количество информации только увеличивается. Крупнейшие социальные сети, СМИ, исследовательские сообщества ежедневно пополняют интернет новой информацией. Количество информации неуклонно растет и обрабатывать её вручную невозможно, да и человеческих ресурсов потребовалось бы слишком много. Это привело к тому, что человек уже не в состоянии проанализировать множество имеющихся новостных потоков. Разработка методов для автоматической обработки информации позволяет сократить объём информационного потока до разумных размеров. С применением автоматического анализа, человеку не требуется просматривать огромное количество новостных материалов для того, чтобы оставаться в курсе последних событий. Таким образом, задача автоматической обработки новостных статей является востребованной и актуальной. Анализ текстов на естественном языке представляет собой задачу обработки неструктурированной информации. Обнаружением скрытых зависимостей и извлечением полезных сведений из больших объёмов информации занимаются специалисты области data mining. Одним из популярных подразделов data mining является текстовый анализ (text mining). Популярность текстового анализа обусловлена увеличивающимися объемами информации на естественном языке и возможностью их обработки. Text mining производит анализ текстовой информации с помощью математических методов. Типичными задачами интеллектуального анализа текстов являются:

- задачи классификации и кластеризации данных;

- нахождение шаблонов данных;
- построение иерархии объектов;
- определение тематики и тональности текстов;
- автоматическое автореферирование документов;
- извлечение фактов и понятий;
- и многие другие.

Постановка задачи

Целью данной выпускной квалификационной работы является разработка программы, составляющей обзор событий для имеющейся коллекции новостных документов.

Для достижения поставленной задачи требуется:

- Сформировать тестовую коллекцию новостных сообщений
- Получить разбиение коллекции на группы, соответствующие различным событиям
- Составить обзор событий
- Изучить методы машинного обучения, необходимые для решения определенных подзадач
- Изучить используемый инструментарий для разработки (язык программирования Python, библиотеки: scikit-learn, python-igraph, networks, nltk, pymorphy2)
- Сформировать обучающую коллекцию предложений
- Провести тестирование и проанализировать полученный результат

Глава 1. Кластеризация

Как правило, новостные порталы размещают за очень короткое время множество статей, посвященных одному и тому же значимому событию, например, катастрофе или Олимпийским играм. В связи с этим, в первую очередь, требуется объединить новостные публикации, соответствующие одному событию, в группы. Такая задача называется задачей кластеризации.

1.1 Постановка задачи кластеризации [1]

Имеется два множества - X и Y. Первое - множество объектов, второе - множество кластеров. Множество Y в некоторых случаях бывает известно заранее. Однако, чаще всего необходимо определить оптимальное число кластеров, используя тот или иной *критерий качества* кластеризации. Обычно множество X представляет собой n-мерный вектор: $\forall \bar{x} \in X : \bar{x} = (x^1, \ldots, x^n),$ где $x^i \in R^n, \ \forall\ i = \overline{1, n}$

На множестве X задана функция расстояния: $\rho : X \times X \to [0, \infty)$. Функция расстояния должна быть метрикой, т.е. должны выполняться условия:

1) Симметрия: $\rho(x, y) = \rho(y, x)$
2) Тождественность: $\rho(x, y) = 0 \ \Leftrightarrow x = y$
3) Неравенство треугольника: $\rho(x, y) + \rho(y, z) \leq \rho(x, z)$

Требуется найти такую функцию $\alpha : X \to Y$, чтобы каждый кластер состоял из наиболее близких объектов , а объекты разных классов были существенно различны.

Решение задачи кластеризации принципиально неоднозначно [2]:

1) Точной постановки задачи кластеризации не существует.
2) Универсального критерия качества кластеризации не существует, несмотря на существование огромного количества таких критериев.
3) Результаты кластеризации могут существенно отличаться при выборе различных метрик, поскольку метрику ρ эксперт задаёт субъективно.
4) Число кластеров |Y |, как правило, заранее неизвестно, что осложняет процесс разделения объектов на группы.

1.2 Обзор существующих методов кластеризации

Существующие методы кластеризации можно условно разделить на следующие группы:

1) Иерархические методы кластеризации
2) Статистические и вероятностные методы кластеризации (четкие и нечеткие)
3) Графовые методы
4) Подходы на основе искусственного интеллекта (нейронные сети, эволюционные методы)

1.3 Иерархические алгоритмы кластеризации [3]

Алгоритмы иерархической кластеризации представляют собой достаточно широкий класс алгоритмов. Данные методы бывают двух видов, работающие сверху-вниз (когда сначала все элементы рассматриваются как один кластер) и работающие снизу-вверх (когда на этапе инициализации каждый элемент содержится в своём кластере). Однако, обычно используются алгоритмы, работающие снизу-вверх. Кроме того, иерархические алгоритмы отличаются друг от друга

используемой метрикой, которая отражает похожесть групп. Общая схема работы данных алгоритмов выглядит следующим образом:

1) на этапе инициализации каждый элемент принадлежит отдельному кластеру;
2) на каждом последующем шаге, исходя из выбранной метрики, вычисляются попарные расстояния между существующими группами;
3) группы, оказавшиеся наиболее близкими, объединяются в новую;
4) алгоритм завершится, когда останется только одна группа, таким образом будет построена иерархия групп.

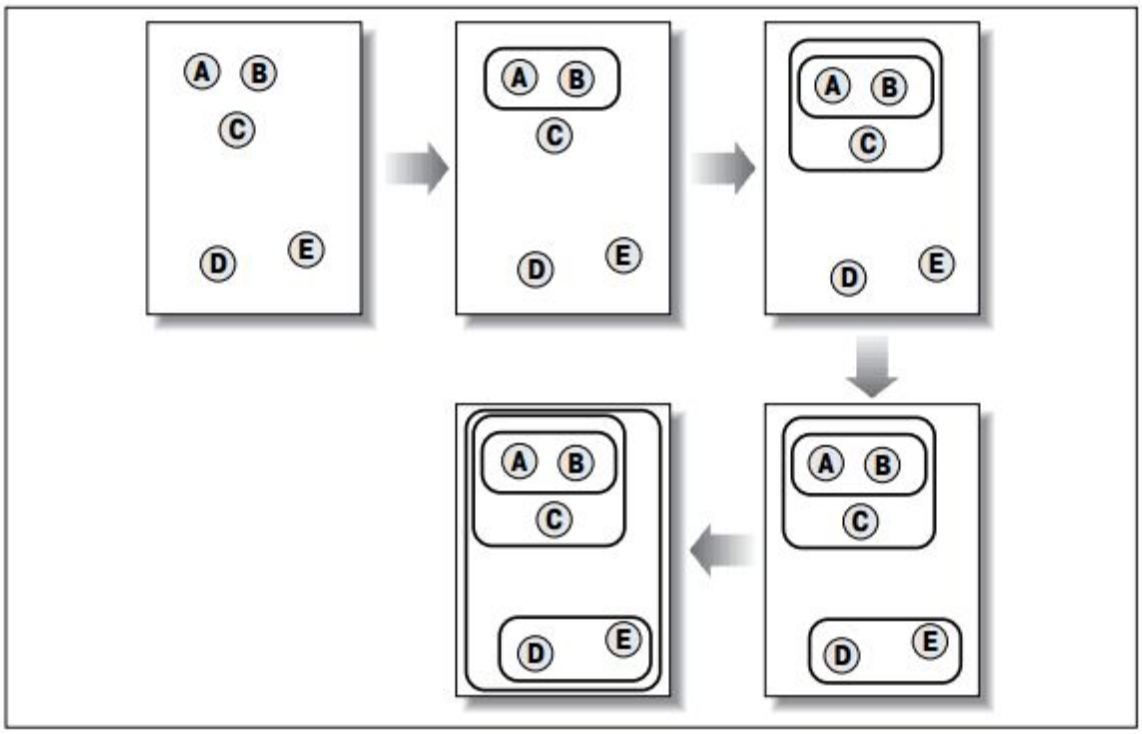

Рис. 1 Иерархическая кластеризация в действии

Результат работы такой кластеризации представляется в виде графа, который называется дендрограммой.

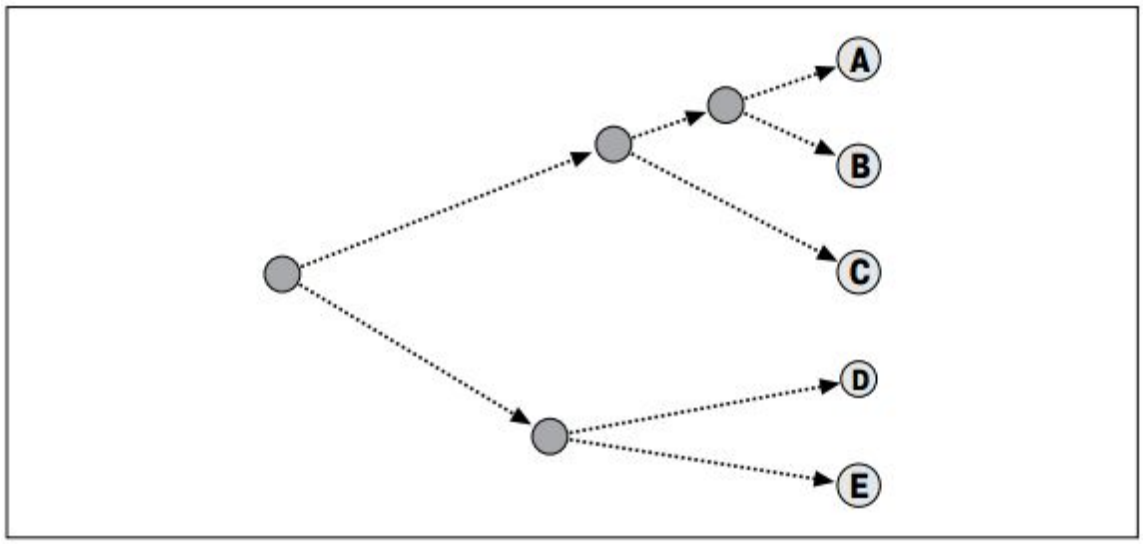

Рис. 2 Дендрограмма - визуализация иерархической кластеризации

Дендрограмма отображает как порядок объединения элементов, так и расстояния между объектами.

Для того чтобы получить разбиение исходного множества элементов на кластеры необходимо вручную задать их число и, исходя из этого, сделать соответствующие сечение бинарного дерева(дендрограммы).

Наиболее популярными иерархическими алгоритмами являются Single/Complete/Average Link [4], для которых межкластерное расстояние определяется следующим образом:

1) минимальное расстояние между парой объектов в соседних кластерах (Single Link);
2) максимальное расстояние между парой объектов соседних кластеров (Complete Link);
3) среднее расстояние между парой любых двух элементов в соседних кластерах (Average Link).

Метод Average Link является компромиссом между Single и Complete Link как по скорости, так и по точности. Метод Single Link работает быстрее - $O(N^2)$, чем Complete Link - $O(N^3)$, где N – число

документов коллекции. Однако Single Link строит слишком «вытянутые» кластеры [5].

1.4 Статистические и вероятностные методы кластеризации

Наиболее популярным алгоритмом, среди вероятностных подходов, является алгоритм k-means (или метод k-средних). Он был почти одновременно изобретён в 1950-х годах двумя математиками Гуго Штейнгаузом и Стюартом Ллойдом [6]. Данный метод является частным случаем общего метода EM(Expectation Maximization).

Классический вариант алгоритма выглядит следующим образом [7]:

1. Необходимо выбрать k точек $C = \{c_1, c_2, \ldots, c_k\}$, которые будут считаться “центрами” кластеров. Начальные центроиды определяются равномерно случайным образом из X.
2. Для каждого $i \in \{1, \ldots, k\}$ задаем кластер C_i как множество точек из X, которые находятся ближе к центру c_i, чем к c_j для всех $j \neq i$. Каждый элемент из множества объектов X относится к одному из k кластеров.
3. Для каждого $i \in \{1, \ldots, k\}$ вычисляем новый центр кластера c_i как центр масс всех точек из кластера C_i: $c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$.
4. Повторять шаги 2 и 3 до тех пор пока кластерные центры не стабилизируются, т.е. все наблюдения будут принадлежать кластеру, которому принадлежали до текущей итерации, или до тех пор пока число итераций не будет равно максимальному числу итераций.

Идея алгоритма заключается в том, что на шагах 2 и 3 он стремится минимизировать суммарное квадратичное отклонение точек кластеров от центров этих кластеров: $\varphi = \sum_{x \in X} min_{c \in C} ||x - c||^2$.

На Рис. 3 показан процесс объединения пяти элементов в два кластера.

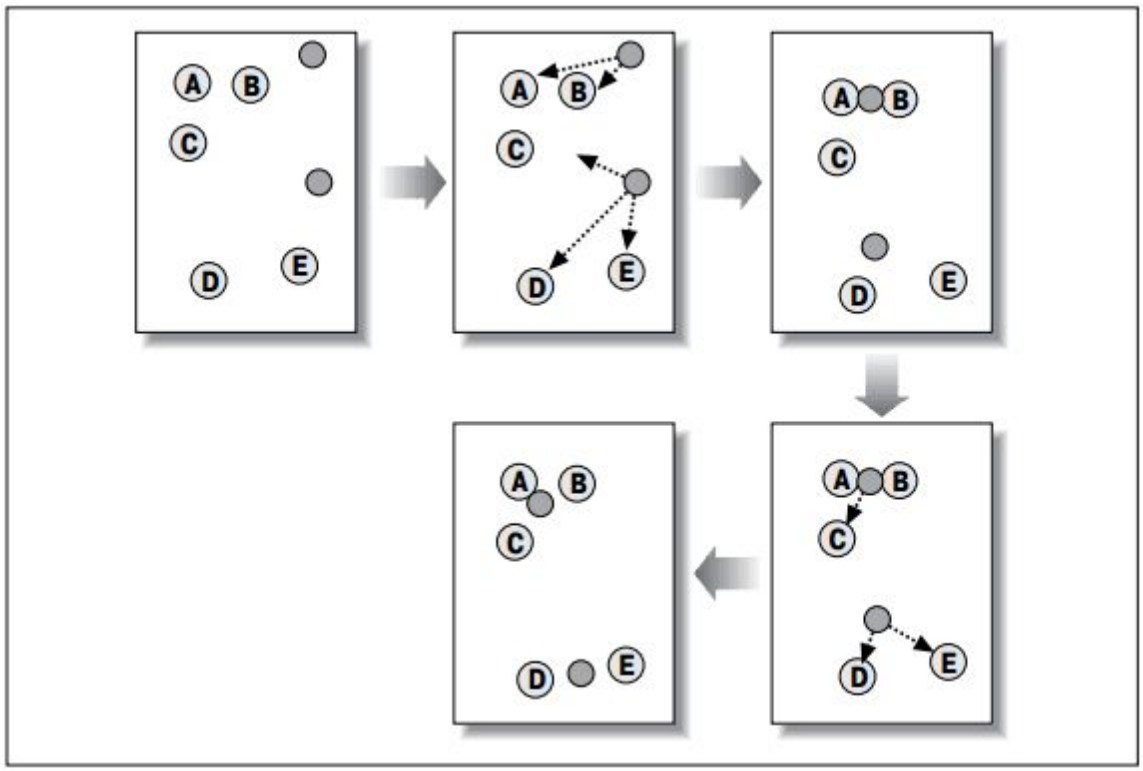

Рис 3. Кластеризация методом k-means с двумя кластерами

K-means отличается понятностью, простотой и быстротой. Средняя сложность $O(knT)$, где k - число кластеров, n - число элементов и T - количество итераций. Однако, алгоритм имеет следующие недостатки:

1) В качестве входного параметра алгоритм требует желаемое число кластеров k.
2) Алгоритм чувствителен к выбору первоначальных центров кластеров.

3) Не гарантируется достижение глобального минимума суммарного квадратичного отклонения φ, а только одного из локальных минимумов.

Также существует улучшенная версия данного алгоритма - k-means++. Её в 2007 году предложили Дэвид Артур и Сергей Вассильвитский [7]. Они предложили конкретный способ выбора начальных центроидов:

1a) Первый центр c_1 выбирается случайным образом из X.

1b) Выбрать новый центр c_i из X с вероятностью $\frac{D(x)^2}{\sum_{x \in X} D(x)^2}$, где $D(x)$ - кратчайшее расстояние от точки до ближайшего центра, который мы уже выбрали. При выборе каждого следующего центроида не нужно специально следить за тем, чтобы он не совпал с одной из уже выбранных в качестве центроидов точек, так как вероятность повторного выбора некоторой точки равна нулю.

1c) Повторять шаг 2 до тех пор, пока не наберётся k центров.

2-4) Аналогичны соответствующим шагам в классической реализации.

1.5 Графовые методы кластеризации

Наиболее широким классом, не требующим задания предполагаемого количества кластеров, является класс графовых алгоритмов кластеризации.

Данные алгоритмы предполагают представление исходных данных в виде графа $G = (V, E)$, вершинами которого являются объекты, а ребра имеют вес, равный расстоянию между ними.

Рассмотрим такие графические методы кластеризации как Walktrap и Infomap. Оба метода основаны на случайных блужданиях. Выбор упомянутых методов обоснован тем, что лексика, используемая в новостных текстах не отличается разнообразием, что затрудняет процесс разделения на кластеры. По этой причине в группы объединяются тексты, имеющие в исходной выборке порядковые номера, идущие друг за другом. Автор предполагает, что процесс случайного блуждания позволит объединять в кластеры объекты вне зависимости от их расположения.

1.5.1. Метод Walktrap [8]

Граф G ассоциируется с его матрицей смежности $A : A_{ij} = 1$, если вершины i и j имеют смежное ребро, в противном случае $A_{ij} = 0$. Степенью вершины i будем называть величину $d_i = \sum_j A_{ij}$.

В алгоритме Walktrap используется идея случайных блужданий. На каждом шаге процесса блуждающий объект находится в вершине, из которой он равновероятным образом перемещается в соседние вершины. Последовательность таких посещений вершин составляет цепь Маркова, состояниями которой являются вершины исходного графа. На каждом шаге вероятность перехода из вершины i в вершину j будет равна $P_{ij} = \frac{A_{ij}}{d_i}$. Таким образом определяется матрица P переходных вероятностей процесса случайного блуждания. Матрицу P можно представить в виде $P = D^{-1}A$, где D - матрица, на диагонали которой стоят степени вершин.

С помощью матрицы переходных вероятностей можно получить вероятность перехода из вершины i в вершину j на t шагов. Для этого необходимо вычислить $(P^t)_{ij}$. Для краткости далее будем опускать скобки

P_{ij}^t.

Для того, чтобы сгруппировать вершины, необходимо определить способ сравнения близости вершин. Считается, что если расстояние между двумя вершинами велико, то они принадлежат различным сообществам (кластерам), иначе они являются членами одного кластера. Определим расстояние между вершинами как:

$$r_{ij} = \sqrt{\sum_{k=1}^{n} \frac{(P_{ik}^t - P_{jk}^t)^2}{d_k}} = \|D^{-\frac{1}{2}} P_{i\bullet}^t - D^{-\frac{1}{2}} P_{j\bullet}^t\|,$$

где $\|\bullet\|$ - евклидова норма, $P_{i\cdot}^t$ - вектор столбец, в j-ой позиции которого стоит значение P_{ij}^t. Следует заметить, что величина r_{ij} зависит от параметра t.

Теперь определим расстояние между самими сообществами. Пусть $C_1,\ C_2 \subset V$, тогда расстояние между сообществами определяется как:

$$r_{C_1C_2} = \|D^{-\frac{1}{2}} P_{C_1\bullet}^t - D^{-\frac{1}{2}} P_{C_2\bullet}^t\| = \sqrt{\sum_{k=1}^{n} \frac{(P_{C_1k}^t - P_{C_2k}^t)^2}{d_k}},$$

где $P_{Cj}^t = \frac{1}{|C|} \sum_{i \in C} P_{ij}^t$, а $P_{C\bullet}^t$ - вектор из вероятностей P_{Cj}^t.

Теперь, когда метрика определена, задача выделения сообществ сведена к задаче кластеризации вершин. Используется подход, основанный на методе Варда[9].

Укажем алгоритм объединения вершин в кластеры.

1. Инициализация. Каждая вершина содержится в отдельном кластере, т.е. начальное разбиение выглядит как $P_1 = \{\{v\},\ v \in V\}$.
2. Вычисление расстояния между всеми смежными вершинами.
3. На каждом шаге k.

   (а) Выбрать два сообщества C_1 и C_2 из P_k по критерию, основанному на метрике, которая будет описана ниже.

(б) Объединить эти сообщества в одно новое, $C_3 = C_1 \bigcap C_2$.

Соответствующим образом преобразовать P_k в P_{k+1} .

(в) Обновить расстояния между сообществами.

На $n-1$ шаге мы получаем $P_n = \{V\}$ и алгоритм завершается. Таким образом получена дендрограмма для вершин графа т.е. бинарное дерево, отражающие порядок объединения вершин в кластеры.

Объединение сообществ происходит согласно методу Варда. Таким образом рассматриваются только те пары сообществ C_1 и C_2, которые инцидентны друг другу. На каждом шаге k два сообщества объединяются так чтобы значение σ_k было минимально. Где σ_k - квадрат расстояния между каждой вершиной и их сообществом: $\sigma_k = \frac{1}{n} \sum_{C \in P_k} \sum_{i \in C} r_{iC}^2 \rightarrow min$.

Такая задача эквивалентна поиску C_1 , C_2 минимизирующих величину: $\Delta\sigma(C_1, C_2) = \frac{1}{n}(\sum_{i \in C_3} r_{iC_3}^2 - \sum_{i \in C_1} r_{iC_1}^2 - \sum_{i \in C_2} r_{iC_2}^2) \rightarrow min_{C_1, C_2}$.

Сложность метода $O(n^2 log(n))$.

1.5.2. Метод Infomap [10]

Как и предыдущий алгоритм, Infomap использует механизм случайных блужданий. Здесь задача кластеризации определяется как задача кодирования пути, который пройдет блуждающий элемент, длину которого пытаются минимизировать.

Каждое сообщество, а также каждая вершина в нем имеют свой уникальный бинарный код. Также есть дополнительный код выхода из сообщества, не совпадающий с кодами вершин в этом сообществе.

При попадании в другое сообщество записывается его код и внутренний код вершины, в которую попал объект. При переходах внутри сообщества записываются внутренние коды вершин. Если осуществляется переход в другое сообщество пишется код выхода из данного сообщества и код нового.

Т.е. имеется два уровня кодирования: уровень сообществ и вершин. Для каждого из них используются коды Хаффмана [11].

В таком виде среднее описание длины одного перехода равно:

$$L(M) = qH(C) + \sum_{i=1}^{m} p^i H(C_i),$$

где q - вероятность покинуть какое-либо сообщество на любом шаге, $H(C)$ - энтропия кодов сообществ, $H(C_i)$ - энтропия кодов внутри сообщества C_i, вес p^i - это доля перемещений внутри сообщества C_i сложенная с вероятностью покинуть сообщество.

Матрица P, описанная в предыдущем методе, позволяет вычислять вероятность посещения той или иной вершины. На основе этой информации происходит запуск жадной оптимизации функционала $L(M)$.

1.6. Метод главных компонент

Очень часто процесс кластеризации затрудняется тем, что матрица исходных данных $X_{m\times n}$ (матрица “объекты-признаки”) имеет слишком большую размерность, что приводит к большой вычислительной сложности. К тому же, если в обучающей выборке между признаками есть корреляционная зависимость (мультиколлинеарность), то это может привести к неприятным последствиям: получение неустойчивых оценок

параметров модели, невозможности правильно оценить значимость параметров модели.

Основная идея метода метода главных компонент - обеспечить такое линейное преобразование признаков, чтобы избавиться от корреляционной зависимости. Кроме этого, метод главных компонент или Principal component analysis (PCA) [12] - один из основных способов уменьшить размерность данных. PCA был предложен Карлом Пирсоном в 1901 году. Существует несколько способов реализации метода:

1) Вычисление собственных векторов и собственных значений матрицы исходных данных.
2) Сингулярное разложение центрированной матрицы исходных данных.
3) Алгоритм NIPALS(Nonlinear Iterative Partial Least Squares) для первых k компонент.

В классической реализации матрица исходных данных представляется в виде $X = TP^T$, где T - ортогональная матрица счетов(score matrix) , столбцы t_i которой - главные компоненты, P - ортогональная матрица нагрузок(loadings matrix). Сокращение размерности происходит за счёт взятия первых k столбцов матриц T и P: $X = T_k P_k^T + E$, где E - матрица невязок.

Для того чтобы определить матрицу T необходимо построить матрицу ковариаций столбцов матрицы X:

$$C = cov(X, Y) = E[(X - EX)(Y - EY)^T]$$

$$C = \{c_{ij}\},\ c_{ij} = \frac{1}{m-1} \sum_{l=1}^{m} (x_{li} - \overline{X_i})\,(x_{lj} - \overline{X_j}),$$

где $\overline{X_i}$ и $\overline{X_j}$ - средние соответствующих компонент векторов.

Затем необходимо найти собственные векторы t_i и собственные числа λ_i матрицы C. Матрица T формируется из столбцов t_i, отсортированных по убыванию значений соответствующих λ_i ($\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n \geq 0$).

Если матрица исходных данных центрирована т.е. $x_{ij} = x_{ij} - \overline{x_j}, \overline{j = 1, n}$, то такая задача эквивалентна нахождению сингулярного разложения матрицы X [13].

Определение. Пусть A - произвольная вещественная матрица размерности $m \times n$ $(m \geq n)$. Число $\sigma \in R : \sigma \geq 0$ называется сингулярным числом матрицы A тогда и только тогда, когда существуют два вектора единичной длины: $u(m \times 1)$ и $v(n \times 1)$ такие, что: $Av = \sigma u$ и $A^T u = \sigma v$. Вектор u называется левым сингулярным вектором, а v - правым сингулярным вектором.

Сингулярным разложением матрицы A называется её представление в виде

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

где U - ортогональная матрица размерности $m \times m$, V - ортогональная матрица размерности $n \times n$ и Σ - диагональная матрица размерности $m \times n$.

Столбцы U - левые сингулярные векторы. Столбцы V - правые сингулярные векторы. Элементы Σ - сингулярные числа матрицы A:

$$\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > \sigma_{r+1} = ... = \sigma_n = 0,$$

где $r = rank(A)$, $r \leq n$.

Правые сингулярные вектор-столбцы, участвующие в данном разложении, являются векторами главных компонент и собственными

векторами ковариационной матрицы C, отвечающими положительным собственным числам λ_i.

Таким образом, матрицу исходных данных можно представить в виде $X = U\Sigma V^T$. Получаем, что $T = U\Sigma$, а $P = V$.

Не взирая на то, что задача сингулярного разложения матрицы исходных данных формально совпадает с задачей разложения ковариационной матрицы, алгоритмы вычисления сингулярного разложения напрямую, без вычисления ковариационной матрицы, считаются более эффективными и устойчивыми.

Глава 2. Составление обзора событий

2.1. TextRank

Для анализа текстов на естественном языке активно применяются графовые модели. Их популярность обусловлена тем, что алгоритмы этой группы методов не зависят от языка, на котором написан анализируемый текст.

Графовый алгоритм TextRank [14] является приложением алгоритма PageRank к задачам обработки естественного языка. PageRank - это алгоритм ссылочного ранжирования, который определяет рейтинг каждой вершины основываясь на количестве входящих ребер и их рейтинга. Первоначально TextRank использовался в задачах извлечения ключевых слов и автореферирования для английского языка. Однако, позднее оказалось, что он эффективен и для русского языка [15].

Таким образом, для извлечения наиболее значимых предложений, сперва необходимо представить исходный текст в виде графа $G = (V, E)$. Вершинами такого графа могут быть как отдельные термины, так и целые предложения. Первый вариант применяется, если из текста необходимо извлечь ключевые слова, второй - для выделения основного содержания. В рассматриваемой в работе вершинами графа являются - предложения исходного текста. Стоит отметить, что наиболее важными считаются такие предложения, которые содержат информацию из нескольких других предложений. При этом вес рёбер отражает связь соответствующей пары предложений.

После генерации графа требуется вычислить значение TextRank -

величину стационарного распределения случайного блуждания для каждой вершины $v \in V$ с учётом весов связей:

$$WS(v_i) = (1-d) + d \sum_{v_j \in In(v_i)} \frac{w_{ij}}{\sum_{v_k \in Out(v_j)} w_{jk}} WS(v_j),$$

где d - фактор затухания, $In(v)$ - множество вершин, входящих в v, $Out(v)$ - множество вершин, исходящих из v, w_{ij} - вес ребра (v_i, v_j). Для неориентированного графа $In(v) \equiv Out(v)$.

После вычисления значения TextRank остается составит множество C, состоящие из кандидатов в наиболее значимые предложения в тексте.

Таким образом, метод можно разбить на три основных этапа:

1) построение графа на основе исходного текста на естественном языке;
2) вычисление значения PageRank для построенного графа;
3) применение полученных весов вершин для извлечения сведений из текста.

2.2. Алгоритмы, основанные на деревьях принятия решений

Задачу составления саммари можно также рассматривать как задачу классификации.

2.2.1. Постановка задачи классификации

Пусть имеется некоторое множество описаний объектов X и конечное множество номеров классов Y. Существует неизвестная целевая

зависимость $y^* : X \to Y$, значения которой известны на объектах конечной обучающей выборки $X^m = \{(x_1, y_1), \ldots, (x_m, y_m)\}$. Требуется построить алгоритм $a : X \to Y$, способный относить произвольный объект $x \in X$ к одному из классов множества Y.

Таким образом, получается задача классификации с двумя классами: включать предложение в саммари или не включать.

2.2.2. Общая схема работы методов, основанных на деревьях решений

Наиболее популярными алгоритмами классификации являются деревья принятия решений. Дерево принятия решений - это дерево, в узлах которого располагаются атрибуты, используемые для дифференциации объектов классификации. В листьях помещаются значения целевой функции, а на рёбрах - значения атрибута, от которого зависит целевая функция. Для классификации нового случая необходимо только определить соответствующий путь по дереву до листа и получить соответствующее значение целевой функции.

Предложенный метод обладает следующими достоинствами:

- деревья решений просты для понимания, кроме того их легко интерпретировать;
- метод не требует предварительной подготовки данных, вследствие чего, позволяет работать с большим объёмом информации без подготовительных процедур;
- метод позволяет одновременно работать как с категориальными, так и с интервальными переменными;

- полученную модель можно оценить с помощью статистических методов;
- дерево принятия решений является надежным методом, поскольку достойно справляется с классификацией даже в случае нарушений первоначальных предположений, включенных в модель.

При всех достоинствах данного подхода стоит учитывать тот факт, что велика вероятность переобучения модели. Достаточно просто создать слишком большую конструкцию, представляющую данные не полностью. Для того чтобы избежать переобучения классификатора, необходимо регулировать глубину дерева. Кроме этого существует и другой недостаток - проблема построения оптимального дерева.

Общая схема построения дерева выглядит следующим образом:

- выбирается очередной атрибут Q и помещается в корень;
- Для всех значений атрибута i:
    - из тестовых примеров выбираются те, у которых значение атрибута Q равно i;
    - рекурсивно строится дерево для данного потомка;
- получается построенное дерево.

Существует множество алгоритмов, построения дерева принятия решений. Самые популярные среди них: ID3, C4.5, CART. Эти методы отличаются друг от друга только способом выбора очередного атрибута.

2.2.3. Алгоритм ID3

Алгоритм ID3 [16] был предложен Джоном Р. Квинланом. Выбор очередного атрибута происходит на основании прироста

информации(Gain):

$$Gain(A,Q) = H(A,S) - \sum_{i=1}^{q} \frac{|A_i|}{|A|} H(A_i,S),$$

где A - множество элементов, часть из которых обладает свойством S, классифицированного посредством атрибута Q, имеющего q возможных значений; A_i - множество элементов A, на которых атрибут Q имеет значение i; $H(A,S)$ - энтропия множества A по отношению к свойству S. Энтропия определяется следующим образом

- если свойство бинарное: $H(A,S) = -\frac{m}{n} log_2 \frac{m}{n} - \frac{n-m}{n} log_2 \frac{n-m}{n}$, где n - число элементов множества A, а m - число элементов множества A, обладающих свойством S;
- если свойство S может принимать s различных значений, каждое из которых реализуется в m_i случаях: $H(A,S) = -\sum_{i=1}^{s} \frac{m_i}{n} log \frac{m_i}{n}$.

Т.е. на каждом шаге алгоритм должен выбирать тот атрибут, для которого прирост информации максимален. Однако у критерия прироста информации имеется один существенный недостаток. Такой критерий будет выбирать те атрибуты, которые имеют больше всего значений. В результате чего Квинлан создал усовершенствованную версию ID3 — алгоритм C4.5. Кроме этого ID3 никак не борется с переобучением. Поэтому построенное дерево необходимо дополнительно прореживать.

2.2.4. Алгоритм C4.5

Алгоритм C4.5 [16] использует усовершенствованный критерий Gain Ratio:

$$GainRatio(A, Q) = \frac{Gain(A,Q)}{SplitInfo(A,Q)},$$

где $SplitInfo(A, Q) = -\sum_{i=1}^{q} \frac{|A_q|}{|A|} log_2 \frac{|A_q|}{|A|}$.

Критерий учитывает как количество информации, необходимое для записи результата,так и количество информации, требуемое для разделения по текущему атрибуту. Метод C4.5 использует однопроходный метод прореживания для уменьшения вероятности переобучения.

2.2.5. Алгоритм CART

Алгоритм CART [17] разработан совместно профессорами из Стэнфорда и Беркли. Он строит исключительно бинарное дерево. Здесь в качестве оценки качества модели используется индекс Гини (неопределенность Гини). Если набор классов A содержит данные n классов, тогда индекс Гини определяется следующим образом

$$Gini(A) = 1 - \sum_{i=1}^{n} p_i^2,$$

где p_i - вероятность класса i в A. Если набор A разбивается на две части A_1 и A_2 с числом примеров в каждом N_1 и N_2, тогда показатель качества разбиения будет равен

$$Gini_{split}(A) = \frac{N_1}{N} Gini(A_1) + \frac{N_2}{N} Gini(A_2).$$

Наилучшим считается разбиение A значение $Gini_{split}(A)$ для которого минимально. Таким образом, если N - число примеров в узле предке, L и R - число потомков в левом и правом поддеревьях, а l_i и r_i - число экземпляров $i-$ого класса в левом и правом потомках, то качество

разбиения оценивается следующим образом:

$$Gini_{split} = \frac{L}{N}\left(1 - \sum_{i=1}^{n}\left(\frac{l_i}{L}\right)^2\right) + \frac{R}{N}\left(1 - \sum_{i=1}^{n}\left(\frac{r_i}{L}\right)^2\right) \rightarrow min\,.$$

Что эквивалентно

$$Gini_{split} = \frac{1}{L}\sum_{i=1}^{n} l_i^2 + \frac{1}{R}\sum_{i=1}^{n} r_i^2 \rightarrow max\,.$$

Лучшим считается то разбиение, для которого величина критерия максимальна. Таким образом, при построении «дерева решений» по методу CART ищется такой вариант ветвления, при котором максимально уменьшается значение показателя $Gini_{split}(A)$.

Для уменьшения вероятности переобучения используется механизм отсечения дерева при прореживании. Начиная с листьев дерева, CART оценивает ошибку классификации в узле и вне узла. Если погрешность превышает граничную, то ветка отбрасывается.

Глава 3. Практическая часть

Для решения поставленной задачи был выбран язык программирования Python. Выбор обоснован тем, что для данного языка реализовано много библиотек, которые в значительной мере облегчают задачу программирования: pymorphy2, nltk, scikit-learn.

3.1. Кластеризация

При рассмотрении поставленной задачи была обнаружена проблема, связанная со спецификой разных новостных порталов: новости на разных порталах подразделяются на темы. Причем набор тем для разных порталов различен. Поэтому, в первую очередь, требуется объединить новостные документы, соответствующие одной тематике, в группы. Такая задача называется задачей кластеризации. В рассматриваемом случае задача распределения элементов на кластеры имеет ряд особенностей. Во-первых, число кластеров невозможно задать заранее поскольку обрабатывается произвольная выборка новостей. Во-вторых, новостные сообщения не отличаются разнообразием используемой лексики, что делает тексты лексически очень близкими друг к другу. В-третьих, разбиение на группы осложняется особенностями русского языка. Например, два однокоренных слова могут употребляться в качестве синонимов, в то время как при автоматической обработке текстов эти слова воспринимаются как два разных слова.

Кластеризация проводится для текстов русскоязычных новостных лент. Перед тем как приступить к разбиению на группы, необходимо подготовить данные. Предварительная обработка включает в себя следующие этапы:

1) Извлечение (pdf,html,…)

   В качестве тестовой выборки были взяты новостные тексты с портала lenta.ru [18]. Для выгрузки данных использовались библиотеки lxml.html и requests.

2) Разбиение на слова и предложения (tokenization)

   Токенизацию будем проводить с помощью библиотеки nltk(Natural Language Toolkit), которая позволяет достаточно эффективно разбивать тексты, представленные на естественном языке, на слова и предложения. Кроме того, упомянутая библиотека поддерживает русский язык.

3) Для улучшения качества кластеризации необходимо очистить исходные данные от стоп-слов т.е. служебных частей речи: союзов, предлогов, частиц . Они содержатся в каждом новостном сообщении и не являются уникальными словами, определяющими суть текста. Таким образом, наличие стоп-слов привело бы к ненужному увеличению матрицы исходных данных. Для нахождения стоп-слов используется библиотека pymorphy2, которая позволяет определять части речи слова.

4) Лемматизация или стемминг.

   Обе эти процедуры позволяют приводить различные словоформы к одному исходному виду. С учётом особенностей русского языка, такая процедура становится особенно актуальной. Так, например, в двух текстах может содержаться описание одного и того же события, причем название события будет встречаться в текстах в разных падежах. Обычное посимвольное сравнение двух строк привело бы к определению таких слов как двух совершенно разных. Таким образом, стемминг и лемматизация позволяют считать две

различные формы слова за одно слово. Это также позволяет не учитывать лишние данные. Лемматизация осуществляет такой учёт посредством приведения исходного слова к его лемме т.е. нормальной словарной форме, а стемминг усекает слово до его основы. Для приведения слова к его лемме будем использовать уже знакомую библиотеку pymorphy2. Алгоритм, используемый при лемматизации описан в статье [19]. Для стемминга используется метод RussianStemmer() из библиотеки nltk.stem.snowball. Данный стеммер использует алгоритм Портера для усечения слова [20][21].

5) Представление данных в виде матрицы $tf - idf$ [22], строки которой соответствуют номеру документа, а столбцы - терминам, входящим во все документы.

TF (*term frequency*) - отношение числа вхождений некоторого слова к общему числу слов в документе. С помощью частоты слова t_i оценивается его важность в пределах рассматриваемого документа.

$$tf(t,d) = \frac{n_t}{\sum_k n_k},$$

где в числителе n_t - количество вхождений слова t в документ d, а в знаменателе находится общее число слов данного документа.

IDF (*inverse document frequency*) - обратная частота, с которой слово встречается во всех документах коллекции. Учёт IDF уменьшает вес наиболее употребляемых слов.

$$idf(t,D) = log\frac{|D|}{|\{d_i \in D | t \in d_i\}|},$$

где

- |D| — число документов в коллекции;

- $\left|\{d_i \in D | t \in d_i\}\right|$ — число документов из множества всех документов D, в которых встречается t (когда $n_t \neq 0$).

Таким образом, мера TF-IDF определяется произведением двух сомножителей:

$$tf - idf(t, d, D) = tf(t, d) \times idf(t, D)$$

Большое значение меры TF-IDF получают слова, имеющие высокую частоту употребления в пределах конкретного документа и низкую в остальных документах.

На этом процесс предварительной подготовки данных завершен. Полученные тексты можно кластеризовать.

Методы иерархической кластеризации слабо применимы к рассматриваемой задаче. Во-первых, если количество документов будет слишком большим, то время работы иерархического алгоритма будет несоизмеримо велико. Во-вторых, такие методы требуют знания желаемого числа кластеров. В-третьих, требуется прилагать дополнительные усилия для извлечения кластеров из дендрограммы.

Для решения поставленной задачи был рассмотрен класс графовых алгоритмов кластеризации. Главное преимущество таких методов в том, что они не требуют априорных знаний об изначальной выборке, что в исследуемом случае является актуальным. Из всего множества графовых алгоритмов были выбраны методы Walktrap и Infomap, основанные на процессе случайных блужданий. Выбор методов обоснован тем, что лексика, используемая в новостных текстах не отличается разнообразием. Это затрудняет процесс разделения на кластеры. В результате, в группы, как правило, объединяются тексты, имеющие в исходной выборке близкие порядковые номера. Предполагается, что процесс случайного

блуждания позволит объединять в кластеры объекты вне зависимости от их расположения.

Для представления исходных данных в виде графа был использован пакет python-igraph. Кроме того, в этом пакете реализованы рассматриваемые алгоритмы. Вершинами полученного графа будут являться документы из коллекции, а ребра будут иметь вес равный расстоянию между документами. Поскольку новостные тексты не отличаются разнообразием употребляемых слов, то при использовании Евклидовой метрики, расстояние между любыми документами будет примерно одинаковым. Поэтому вместо евклидового расстояния используем расстояние Чебышева: $\rho(x, x') = max(|x_i - x_i'|)$.

После применения алгоритма Walktrap к коллекции, состоящей из тридцати одного документа, в качестве результата кластеризации была получена дендрограмма, приведенная на рисунке 4.

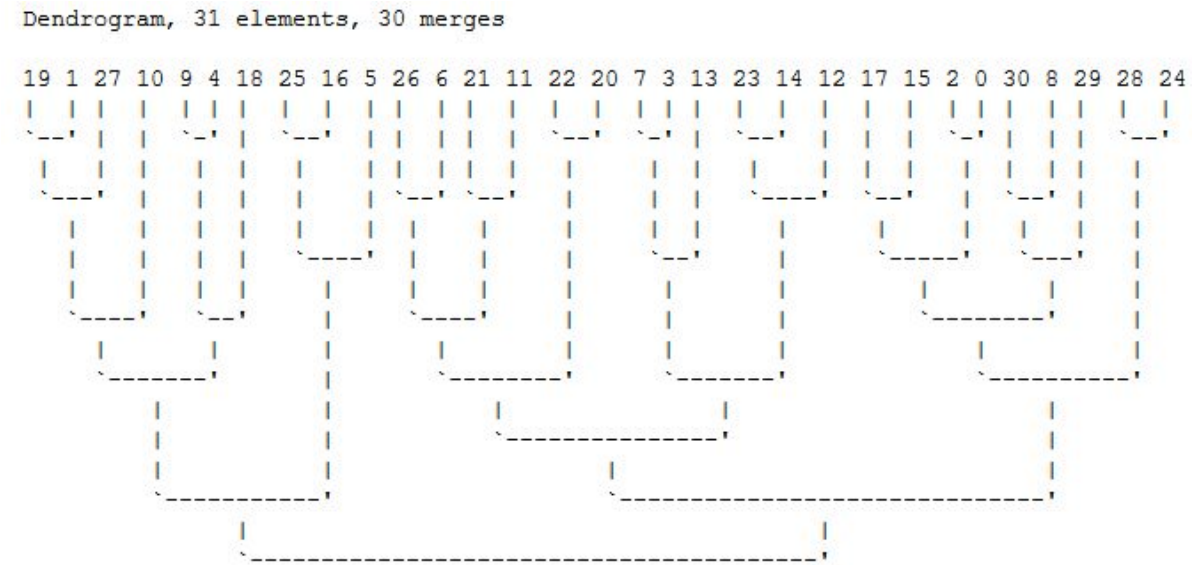

Рис. 4. Результат работы алгоритма Walktrap

В качестве результата работы метода Infomap получается конечное разбиение на кластеры. Графическое представление результата, полученного на том же наборе данных приведено на рисунке 5.

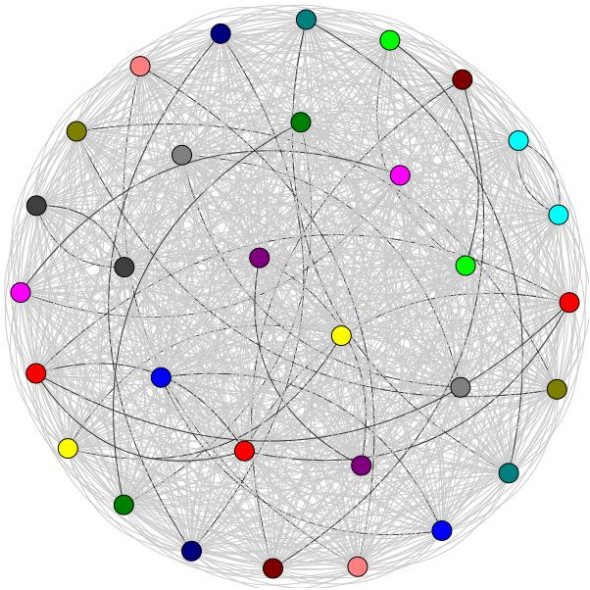

Рис.5. Результаты работы алгоритма Infomap

Вершины(документы), объединенные в один кластер отмечены на рисунке 5 одним цветом.

Далее необходимо каким-либо образом оценить качество кластеризации. Большинство существующих методов оценки основываются на наличие правильного разбиения исходных данных. Однако, в действительности в поставленной задаче нет этой информации. Оценивать эффективность графовых алгоритмов без знания о правильном разбиении, принято с помощью функционала модулярности

$$Q = \frac{1}{2m}\sum_{i,j}(A_{ij} - \frac{d_i d_j}{2m})\delta(C_i, C_j),$$

где $\delta(C_i, C_j) = 1$ если $C_i = C_j$, иначе $\delta(C_i, C_j) = 0$.

Значение модулярности равно разности между долей ребер внутри сообщества и ожидаемой долей связей при случайном размещении рёбер. Большие значения модулярности соответствует более высокому качеству кластеризации.

Недостатком данного способа оценки результатов является то, что функционал не учитывает масштаба данных, в результате чего маленькие сообщества могут быть не учтены при оценивании. Однако, существует поправка к данному функционалу, которая будет учитывать и маленькие сообщества тоже, о ней можно прочитать в [23].

В таблице 1 представлен результат сравнения описанных выше методов. Для алгоритма Walktrap в качестве полученных кластеров взят срез первого уровня полученной дендрограммы.

	Walktrap				Infomap			
Количество документов	31	298	31	298	31	298	31	298
Значение модулярности	-0.034 25	**-0.003 337**	-0.035 708	-0.0033 43	**-0.031 078**	-0.00 3344	-0.0352 39	-0.003 355
Количество кластеров	23	148	20	191	15	149	28	297
	стемминг		лемматизация		стемминг		лемматизация	

Таблица 1. Результат работы графовых методов

Как видно из таблицы, для малых данных лучший результат модулярности показал метод Infomap со стеммингом, а для данных большего размера - Walktrap в комбинации со стеммингом. Тем не менее, результаты, получаемые при использовании стемминга и лемматизации несущественно отличаются друг от друга. С другой стороны, стоит обратить внимание на количество выделенных кластеров. Результаты, в которых количество кластеров сравнимо с количеством исходных документов, нельзя считать приемлемыми. Вцелом, оба алгоритма дают одинаковые результаты, которые не удовлетворяют нашим требованиям.

Была предпринята попытка применения для кластеризации наиболее известного статистического метода k-means и его улучшения k-means++ . В библиотеке для машинного обучения scikit-learn имеется реализация обоих алгоритмов. В качестве начального параметра данные методы требуют желаемое количество кластеров. Таким образом, необходимо сделать некоторое предположение об имеющейся коллекции документов. Как правило, большинство новостей описывают различные события и лишь небольшая их часть посвящена одному событию. Поэтому число кластеров можно оценить примерно как 80% от количества исходных данных. Также попробуем улучшить результаты, используя в комбинации с k-means и k-means++ метод главных компонент (PCA).

Получив разбиение исходной коллекции, необходимо оценить качество проведённой кластеризации. Однако, для оценки качества необходимо дополнительно сформировать эталонное разбиение тестовой коллекции на группы. После этого, имея полученное и правильное разбиения, оценим эффективность работы выбранных методов с помощью точности, полноты и F-меры.

Точность и полнота [24]

Точность (precision) и полнота (recall) - это метрики, используемые для оценки алгоритмов извлечения информации.

Точностью в пределах рассматриваемого класса называют отношение количества документов действительно принадлежащих данному классу к количеству документов, которое алгоритм отнёс к нему. Полнота системы – это отношение количества найденных алгоритмом документов, принадлежащих текущему классу, к общему числу документов этого класса в тестовой выборке. Значение точности и полноты легко получить, используя таблицу контингентности:

Класс i		Оценка эксперта	
		Положительная	Отрицательная
Оценка системы	Положительная	TP	FP
	Отрицательная	FN	TN

Таблица 2. Таблица контингентности

Таблица 2 содержит информацию о количестве верных и неверных решений системы для i-ого класса, где:

- TP(True-positive) - истинно-положительное решение;
- TN(True-negative) - истинно-отрицательное решение;
- FP(False-positive) - ложно-положительное решение;
- FN(False-negative) - ложно-отрицательное решение.

Тогда, точность и полноту определяют как:

$$Precision = \frac{TP}{TP+FP}$$
$$Recall = \frac{TP}{TP+FN}$$

F-мера [24]

Разумеется, что чем больше значения полноты и точности, тем лучше качество работы алгоритма. Однако, на практике максимальные значения данных метрик одновременно не достижимы. F-мера является метрикой, которая объединяет в себе информацию о точности и полноте алгоритма. F-мера определяется как взвешенное гармоническое среднее точности и полноты:

$$F = (\beta^2 + 1)\frac{Precision \times Recall}{\beta^2 Precision + Recall},$$

где если $\beta = 1$, то F-мера придаёт одинаковый вес полноте и точности, в

результате чего получается их сбалансированное значение, если $0 < \beta < 1$, то приоритет отдаётся точности, и при значении $\beta > 1$, преимущество предоставляется полноте.

В таблице 3 представлены результаты сравнения статистических методов кластеризации для той же тестовой выборки.

	Precision	Recall	F-measure
k-means	0.7069	0.7241	0.7044
k-means++	0.8621	0.8276	0.8275
k-means with PCA	0.9482	0.9310	0.9343
k-means++ with PCA	0.9827	0.9655	0.9688

Таблица 3. Значения точности, полноты и F-меры для полученного разбиения

Несмотря на актуальность задачи кластеризации данных, на сегодняшний день она окончательно не решена. Универсального алгоритма, позволяющего решать задачи такого плана, не существует. Подтверждение чего получено в данной работе. В качестве попытки улучшения качества кластеризации был применён метод главных компонент к матрице исходных данных. Примение этого метода позволило сократить размерность данных. Предполагается, что останутся только главные компоненты, которые наиболее точным образом описывают исследуемую коллекцию.

Как видно из таблицы методы k-means и k-means++ показывают достойные результаты на тестовой коллекции. Тем не менее, метод

главных компонент существенно повышает эффективность рассматриваемых методов, что делает комбинацию k-means++ и PCA наиболее выигрышной.

3.2. Составление обзора событий

В данной работе под составлением обзора событий новостного документа будем понимать выделение предложений, наиболее полно(широко) описывающих освещаемое событие. Выделение из текста основных предложений, отражающих смысловую информацию, является одной из подзадач задачи извлечения информации из неструктурированного текста, т.е. текста на естественном языке. Такая задача называется задачей автореферирования. В англоязычной литературе для обозначения задачи автореферирования используется термин automatic summarization, а полученную аннотацию называют summary(саммари).

Существует два основных подхода к созданию аннотации: обобщение и извлечение. Методы первой категории сначала анализируют исходный текст, а потом, основываясь на полученных результатах, генерируют новый текст, отражающий содержание анализируемого текста. Однако, на данный момент такие методы развиты слабо, особенно для русского языка. Извлекающие же методы после анализа текста выбирают его наиболее важные части.

Одним из популярных алгоритмов, применяемых для автореферирования, является TextRank. Данный алгоритм предполагает представление данных в виде графа. В контексте составления саммари вершинами такого графа будут предложения рассматриваемого текста. Ребра между каждой парой вершин должны иметь вес, отражающий “похожесть” предложений в смежных вершинах. Таким образом,

необходимо определить меру сходства для двух предложений. В работе не учитываются стоп-слова, а остальные слова приводятся к нормальной форме с помощью лемматизации и стемминга.

Наиболее популярные меры сходства, где a - количество объектов первого множества, b - количество объектов второго множества, c - количество объектов, общих для обоих множеств :

- коэффициент Сёренсена: $similarity = \frac{2c}{a+b}$;
- коэффициент Кульчинского: $similarity = \frac{c}{2}\left(\frac{1}{a} + \frac{1}{b}\right)$;
- коэффициент Отиаи: $similarity = \frac{c}{\sqrt{ab}}$;
- коэффициент Шимкевича-Симпсона: $similarity = \frac{c}{min(a,b)}$;
- коэффициент Брауна-Бланке: $similarity = \frac{c}{max(a,b)}$.

Для представления текста в виде графа использовалась библиотека networks.

Алгоритму TextRank, как и остальным алгоритмам автореферирования, необходимо задавать необходимое количество извлекаемых предложений. Поэтому зададим объём обзора как 70% от количества предложений, составляющих исходный текст.

Оценка качества, составленного обзора.

После того как саммари было составлено, необходимо оценить его качество. Качество аннотации будем оценивать с помощью следующих коэффициентов [25]:

- косинусный коэффициент
- дивергенция Дженсена-Шеннона

Косинусный коэффициент $cos(\theta)$ между векторами $A = (a_1,\ a_2, ..., a_n)$ и $B = (b_1,\ b_2, ..., b_n)$ рассчитывается следующим образом:

$$similarity\ =\ cos(\theta)\ =\ \frac{A \cdot B}{\|A\|_2 \|B\|_2} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{1=1}^{n} a_i^2}\sqrt{\sum_{1=1}^{n} b_i^2}}$$

Данный коэффициент оценивает пространственную ориентацию векторов и принимает значения из диапазона [-1, 1]. Если векторы имеют противоположную направленность, то коэффициент принимает значение равное -1. Если же векторы сонаправлены, то равное 1. Таким образом, в рассматриваемом случае для наиболее близких по содержанию текстов косинусный коэффициент будет принимать значение близкое к единице, т.е. чем лучше качество саммари, тем больше значение коэффициента.

Дивергенция Дженсена-Шеннона является одним из способов оценки похожести двух распределений A и B случайной величины, которая вычисляется по формуле

$$JSD = \tfrac{1}{2} D_{KL}(A||M) + \tfrac{1}{2} D_{KL}(B||M),$$

где D_{KL} - дивергенция Кульбака-Лейбнера, которая может быть вычислена следующим образом $D_{KL}(A||B) = \sum_i A(i) log \frac{A(i)}{B(i)}$, а $M = \tfrac{1}{2}(A + B)$. D_{KL} определена только тогда, когда $B(i) = 0$ влечет $A(i) = 0$. Если $A(i) = 0$, то $i-$ое слагаемое считается равным нулю. Дивергенция Дженсена-Шеннона - симметричная и более сглаженная версия дивергенции Кульбака-Лейбнера, принимающая всегда конечное значение. Дивергенции Кульбака-Лейбнера, в свою очередь, показывает количество потерянной информации, в случае если для приближения

распределения A было использовано распределение B, где A - реальное распределение данных, а B - его аппроксимация. Дивергенция Дженсена-Шеннона принимает значения из диапазона $[0, 1]$. Чем ближе значение к нулю, тем лучше аппроксимация исходного распределения, т.е. тем меньше было потеряно информации.

В таблицах 4.1, 4.2 и 5.1, 5.2 представлены результаты работы TextRank для разных мер сходства и разных способов приведения слов к их нормальным формам. В качестве тестовых данных будем использовать шесть текстов, принадлежащим различным категориям.

Нормальная форма	Лемматизация				
Мера схожести	Сёренсен	Кульчинский	Отиаи	Шимкевич-Симпсон	Браун-Бланке
test1	0.9232	0.9232	0.9232	0.9232	0.9232
test2	0.9146	0.9262	0.9262	0.9262	0.9262
test3	0.8985	0.8985	0.8985	0.8985	0.8985
test4	0.8591	0.8591	0.8591	0.8591	0.8591
test5	0.9397	0.9352	0.9352	0.9352	0.9352
test6	0.9318	0.9318	0.9318	0.9318	0.9318
Среднее значение для меры	0,9111	0,9123	0,9123	0,9123	0,9123

Таблица 4.1. Значения косинусного коэффициента для саммари, полученных с помощью Textrank

Нормальная форма	Стемминг				
Мера схожести	Сёренсен	Кульчинский	Отиаи	Шимкевич-Симпсон	Браун-Бланке
test1	0.9146	0.9146	0.9146	0.9146	0.9305
test2	0.9230	0.9230	0.9230	0.9230	0.9230
test3	0.8514	0.8514	0.8514	0.8985	0.8514
test4	0.8591	0.8591	0.8591	0.8591	0.8591
test5	0.9409	0.9303	0.9409	0.9332	0.9409
test6	0.9318	0.9318	0.9318	0.9318	0.9318
Среднее значение для меры	0,9035	0,9017	0,9035	0,9101	0,9061

Таблица 4.2. Значения косинусного коэффициента для саммари, полученных с помощью Textrank

Нормальная форма	Лемматизация				
Мера схожести	Сёренсен	Кульчинский	Отиаи	Шимкевич-Симпсон	Браун-Бланке
test1	0.083	0.083	0.083	0.083	0.083
test2	0.0785	0.0785	0.0785	0.0785	0.0785
test3	0.0872	0.0872	0.0872	0.0872	0.0872
test4	0.1108	0.1108	0.1108	0.1108	0.1108
test5	0.0893	0.0976	0.0976	0.0976	0.0976
test6	0.0718	0.0718	0.0718	0.0718	0.0718
Среднее значение для меры	0,0868	0,0882	0,0882	0,0882	0,0882

Таблица 5.1. Значения дивергенции Дженсона-Шеннона для саммари, полученных с помощью TextRank

Нормальная форма	Стемминг				
Мера схожести	Сёренсен	Кульчинский	Отиаи	Шимкевич-Симпсон	Браун-Бланке
test1	0.0854	0.0854	0.0854	0.0854	0.068
test2	0.0817	0.0817	0.0817	0.0817	0.0817
test3	0.1151	0.1151	0.1151	0.0872	0.151
test4	0.1108	0.1108	0.1108	0.1108	0.1108
test5	0.0872	0.0923	0.0872	0.0957	0.0873
test6	0.0718	0.0718	0.0718	0.0718	0.0718
Среднее значение для меры	0,092	0,0929	0,092	0,0888	0,0951

Таблица 5.2. Значения дивергенции Дженсона-Шеннона для саммари, полученных с помощью TextRank

Как видно из приведённых таблиц, различные метрики сходства не дают существенного улучшения, однако результаты с лемматизацией оказываются несколько лучше, чем со стеммингом. Если при сравнении приводить слова к нормальной форме с помощью лемматизации, то мера Сёренсена оказывается чуть хуже остальных, при приведении с помощью стемминга лучшей оказывается мера Шимкевича-Симпсона. Поэтому рассмотрим результат работы алгоритма для меры Шимкевича-Симпсона с лемматизацией и стеммингом для первого документа тестовой выборки.

Исходный текст новости:

Двое из трех пострадавших, которые получили наиболее серьезные травмы при взрыве гранаты в компьютерном клубе в дагестанском селении Агвали, прооперированы. Об этом «Интерфаксу» рассказал министр здравоохранения республики Танка Ибрагимов. Он отметил, что операции сделали пострадавшим 10 и 20 лет. Ребенок остается в тяжелом состоянии, осколки попали ему в мозг. «Мы ожидаем прибытия из Москвы главных специалистов — детского хирурга и реаниматолога, которые примут решение — проводить этому ребенку операцию на месте или перевести его в Москву», — рассказал Ибрагимов. Третьему пострадавшему также проводится плановая операция. Ранее стало известно, что троих пострадавших на вертолете Росгвардии доставили в Махачкалу.

24 апреля в селении Агвали Цумадинского района Дагестана боевая ручная граната взорвалась в здании компьютерного клуба. Ее принес 20-летний молодой человек, предположительно, сын хозяина здания. В результате ЧП погиб школьник, еще 11 (по другим данным, 13) человек получили ранения. Первоначально сообщалось, что взрыв произошел в сельской школе. Утверждалось, что восьмиклассник, принесший гранату, был задержан. Он рассказал, что нашел ее на улице и решил, что это муляж. Уголовное дело расследуется по статьям о незаконном обороте оружия и причинении смерти по неосторожности.

Результат работы TextRank с леммatизацией и мерой Шимпевича-Симпсона:

Двое из трех пострадавших, которые получили наиболее серьезные травмы при взрыве гранаты в компьютерном клубе в дагестанском селении Агвали, прооперированы. Он отметил, что операции сделали пострадавшим 10 и 20 лет. Ребенок остается в тяжелом состоянии, осколки попали ему в мозг. «Мы ожидаем прибытия из Москвы главных специалистов — детского хирурга и реаниматолога, которые примут решение — проводить этому ребенку операцию на месте или перевести его в Москву», — рассказал Ибрагимов. Третьему пострадавшему также проводится плановая операция. Ранее стало известно, что троих пострадавших на вертолете Росгвардии доставили в Махачкалу.

24 апреля в селении Агвали Цумадинского района Дагестана боевая ручная граната взорвалась в здании компьютерного клуба. Ее принес 20-летний молодой человек, предположительно, сын хозяина здания. Утверждалось, что восьмиклассник, принесший гранату, был задержан. Он рассказал, что нашел ее на улице и решил, что это муляж.

Результат работы TextRank со стеммингом и мерой Шимпевича-Симпсона:

Двое из трех пострадавших, которые получили наиболее серьезные травмы при взрыве гранаты в компьютерном клубе в дагестанском селении Агвали, прооперированы. Об этом «Интерфаксу» рассказал министр здравоохранения республики Танка Ибрагимов. Он отметил, что операции сделали пострадавшим 10 и 20 лет. «Мы ожидаем прибытия из Москвы главных специалистов — детского хирурга и реаниматолога, которые примут решение — проводить этому ребенку операцию на месте или перевести его в Москву», — рассказал Ибрагимов. Третьему пострадавшему также проводится плановая операция. Ранее стало известно, что троих пострадавших на вертолете Росгвардии доставили в Махачкалу.

24 апреля в селении Агвали Цумадинского района Дагестана боевая ручная граната взорвалась в здании компьютерного клуба. Ее принес 20-летний молодой человек, предположительно, сын хозяина здания. Утверждалось, что восьмиклассник, принесший гранату, был задержан. Он рассказал, что нашел ее на улице и решил, что это муляж.

Нетрудно заметить, что оба саммари отличаются друг от друга одним предложением. Причём, если в первом обзоре это предложение содержит факт, описывающий освещаемое событие , то во втором варианте из дополнительного предложения лишь становится известно, кто сообщил новость порталу, что в действительности не является описанием произошедшего события.

Таким образом, результат работы алгоритма с лемматизацией отвечает нашим требованиям лучше, чем со стеммингом. Однако, в процессе экспертного анализа полученных результатов мы столкнулись с проблемой включения в обзор предложений, не отражающих суть случившегося. Поэтому задачу составления саммари также рассматривалась как задача классификации с двумя классами: включать предложение в саммари или не включать. Классификацию можно проводить с помощью дерева принятия решений. Деверо принятия решений - это метод машинного обучения с учителем, т.е. для его

обучения требуется наличие тестовой выборки. Таким образом, необходимо:

- определить набор атрибутов, которые будут характеризовать каждую новость;
- разметить тестовую выборку.

Предполагается, что такой подход позволит включать в обзор только те предложения, которые включил бы в него человек.

В качестве признаков были взяты следующие критерии:

1) является ли предложение первым;
2) является ли предложение последним;
3) длина больше предложения больше трёх слов;
4) длина предложения меньше двадцати слов;
5) наличие в предложении даты, указанной в виде последовательности из числа и месяца или числа и слова “год”;
6) наличие имени собственного, которое совершает действие;
7) значение pagerank [26], вычисленное в пределах заданного текста, меньше среднего pagerank для всей новости;
8) значение pagerank, вычисленное в пределах заданного текста, больше среднего pagerank для всей новости;

Обоснуем выбранные критерии. В первую очередь, следует отдельно обратить внимание на первое и последнее предложения. Так как в первом предложении, как правило, указывается основной факт, о котором пойдёт речь в дальнейшем тексте новости. А последнее предложение, в большинстве случаев, подводит итог под всем, сказанным ранее. Кроме того, нет никакого смысла включать в саммари слишком короткие(2-5

слов) или наоборот слишком длинные предложения, поскольку в коротких предложениях, как правило, новостные порталы сообщают читателю какой источник первым опубликовал данную новость, а длинные предложения представляют собой перечисление несущественных фактов. Также следует учитывать предложения, в которых некоторое имя собственное (для новостей, по большей части, имена собственные представлены именами людей и названиями организаций) совершает какое-либо действие, потому что такие предложения описывают значимые события. Например, “Владимир Путин подписал постановление...”. По аналогичной причине дополнительный вес приписывается предложениям, содержащим дату (“14 апреля Владимир Путин подписал постановление...”) . Дата обычно указывает на произошедшее в этот день событие. Помимо этого учитывается “важность” предложения в пределах текста с помощью подсчета значения PageRank, которое определяет значимость, исходя из его меры сходства с остальными предложениями новости.

Сложности могут возникнуть только с выделением из предложения объектов, представленных именами собственными, и действий, ими совершенных(фактов). Как было сказано выше, имена собственные представляют собой имена людей и названия организаций. Имена людей в новостных текстах - это, как правило, имя и фамилия человека т.е. два существительных с большой буквы в именительном падеже, расположенные друг за другом и не являющиеся географическими названиями. Для определения частей речи будем, как и раньше, использовать библиотеку pymorphy2. Несмотря на достаточно высокую точностью работы, данная библиотека не всегда точно определяет часть

речи. Большинство таких ошибок проявляется в случае фамилий иностранцев. Например, “Дарио Скудери”, здесь “Дарио” pymorphy2 распознает как существительное с тегом имя, что соответствует действительности, а вот “Скудери” определяется с большей вероятностью как глагол. Поэтому простой поиск двух последовательных существительных с большой буквы является неэффективным. Метод MorphAnalyzer.parse() данной библиотеки позволяет получать все возможные варианты разбора слова, причём для каждого разбора указывается его предположительная точность. Иностранные фамилии разбираются правильно с меньшей вероятностью, поэтому нужно обязательно смотреть наличие тега NOUN(существительное) во всех разборах второго слова с большой буквы. Если соответствующий тег присутствует хотя бы в одном из разборов, то с большой вероятностью были найдены имя и фамилия. С поиском организаций проще, их можно искать как слова с большой буквы, находящиеся в кавычках. Для извлечения действий, совершенных уже найденными именами собственными, необходимо в пределах соответствующего предложения найти глагол, который бы согласовывался с найденным существительным в роде, числе и лице. После проделанных процедур будет получен список, состоящий из имен собственных и совершенных ими действий.

Для каждого предложения вычисляется вектор значений атрибутов. Значение некоторого элемента вектора равно 1, если соответствующий критерий выполняется для рассматриваемого предложения. В противном случае, значение элемента вектора приравнивается к 0. Таким образом, каждое предложение будет характеризоваться бинарным вектором, размерность которого равна восьми.

Обучающая выборка должна представлять собой множество бинарных векторов со значениями, соответствующими рассматриваемым атрибутам. Каждое предложение должно быть отмечено 1, если предложение входит в короткое содержание текста, или 0, если предложение не нужно включать в краткое содержание. Разметка предложений проводилась вручную.

Для построение дерева принятия решений использовался алгоритм CART, его реализация представлена в библиотеке scikit-learn. Сначала обучение проводилось на одном тексте. В результате было получено следующее дерево:

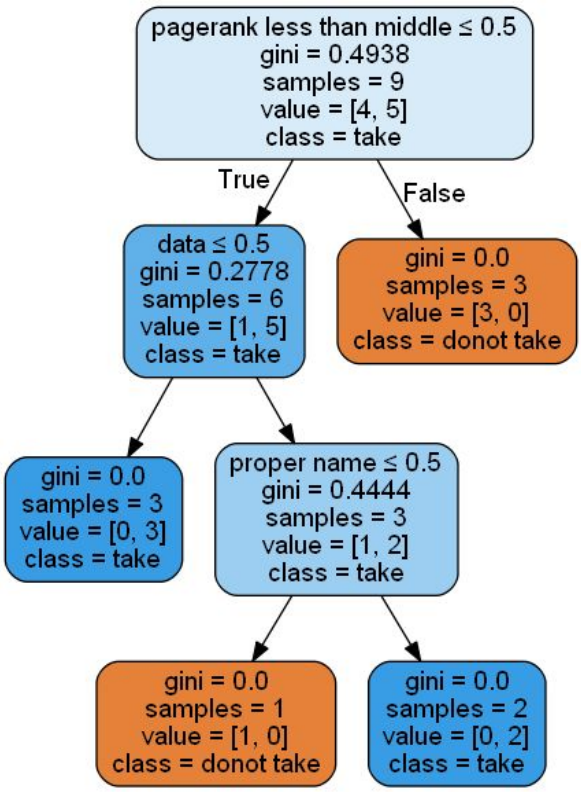

Рис. 6 Дерево принятия решений, полученное при обучении на одном тексте

Далее классификатор был обучен на размеченной обучающей

выборке. В результате получилось дерево, представленное на рисунке 7.

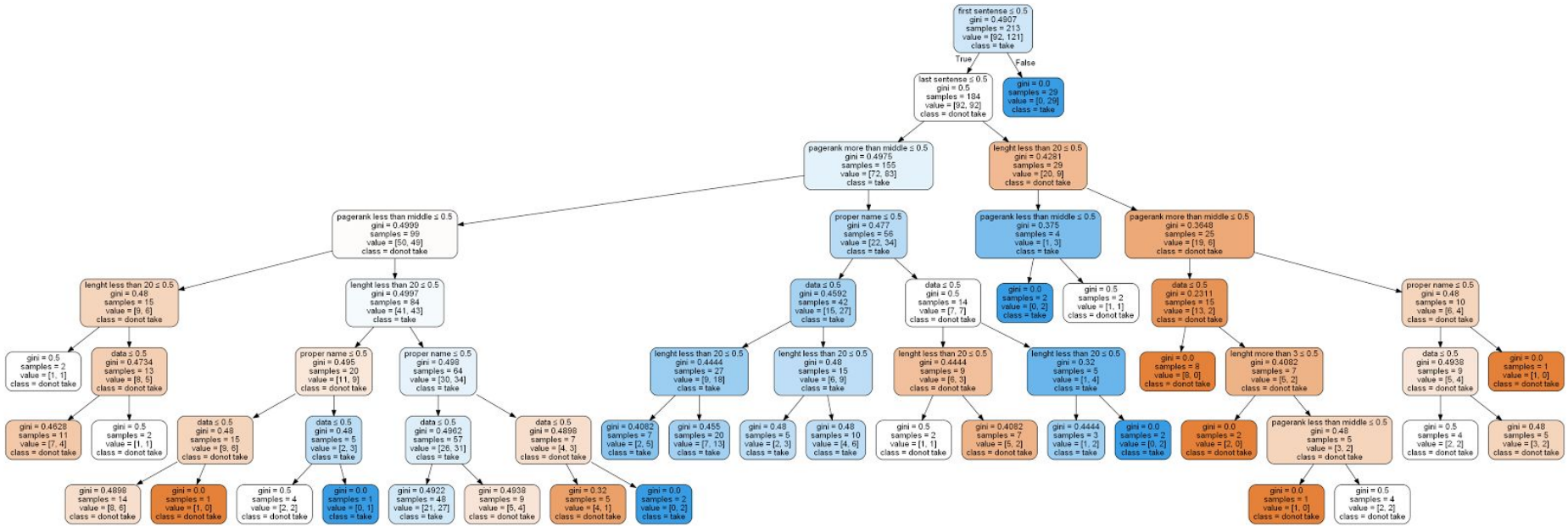
first sentense ≤ 0.5
gini = 0.4907
samples = 213
value = [92, 121]
class = take
True
False
last sentense ≤ 0.5
gini = 0.5
samples = 184
value = [92, 92]
class = donot take
gini = 0.0
samples = 29
value = [0, 29]
class = take
pagerank more than middle ≤ 0.5
gini = 0.4975
samples = 155
value = [72, 83]
class = take
lenght less than 20 ≤ 0.5
gini = 0.4281
samples = 29
value = [20, 9]
class = donot take
pagerank less than middle ≤ 0.5
gini = 0.4999
samples = 99
value = [50, 49]
class = donot take
proper name ≤ 0.5
gini = 0.477
samples = 56
value = [22, 34]
class = take
pagerank less than middle ≤ 0.5
gini = 0.375
samples = 4
value = [1, 3]
class = take
pagerank more than middle ≤ 0.5
gini = 0.3648
samples = 25
value = [19, 6]
class = donot take
lenght less than 20 ≤ 0.5
gini = 0.48
samples = 15
value = [9, 6]
class = donot take
lenght less than 20 ≤ 0.5
gini = 0.4997
samples = 84
value = [41, 43]
class = take
data ≤ 0.5
gini = 0.4592
samples = 42
value = [15, 27]
class = take
data ≤ 0.5
gini = 0.5
samples = 14
value = [7, 7]
class = donot take
gini = 0.0
samples = 2
value = [0, 2]
class = take
gini = 0.5
samples = 2
value = [1, 1]
class = donot take
data ≤ 0.2311
samples = 15
value = [13, 2]
class = donot take
proper name ≤ 0.5
gini = 0.48
samples = 10
value = [6, 4]
class = donot take
gini = 0.5
samples = 2
value = [1, 1]
class = donot take
data ≤ 0.5
gini = 0.4734
samples = 13
value = [8, 5]
class = donot take
proper name ≤ 0.5
gini = 0.495
samples = 20
value = [11, 9]
class = donot take
proper name ≤ 0.5
gini = 0.498
samples = 64
value = [30, 34]
class = take
lenght less than 20 ≤ 0.5
gini = 0.4444
samples = 27
value = [9, 18]
class = take
lenght less than 20 ≤ 0.5
gini = 0.48
samples = 15
value = [6, 9]
class = take
lenght less than 20 ≤ 0.5
gini = 0.4444
samples = 9
value = [6, 3]
class = donot take
lenght less than 20 ≤ 0.5
gini = 0.32
samples = 5
value = [1, 4]
class = take
gini = 0.0
samples = 8
value = [8, 0]
class = donot take
lenght more than 3 ≤ 0.5
gini = 0.4082
samples = 7
value = [5, 2]
class = donot take
data ≤ 0.5
gini = 0.4938
samples = 9
value = [5, 4]
class = donot take
gini = 0.0
samples = 1
value = [1, 0]
class = donot take
gini = 0.4628
samples = 11
value = [7, 4]
class = donot take
gini = 0.5
samples = 2
value = [1, 1]
class = donot take
data ≤ 0.5
gini = 0.48
samples = 15
value = [9, 6]
class = donot take
data ≤ 0.5
gini = 0.48
samples = 5
value = [2, 3]
class = take
data ≤ 0.5
gini = 0.4962
samples = 57
value = [26, 31]
class = take
data ≤ 0.5
gini = 0.4898
samples = 7
value = [4, 3]
class = donot take
gini = 0.4082
samples = 7
value = [2, 5]
class = take
gini = 0.455
samples = 20
value = [7, 13]
class = take
gini = 0.48
samples = 5
value = [2, 3]
class = take
gini = 0.48
samples = 10
value = [4, 6]
class = take
gini = 0.5
samples = 2
value = [1, 1]
class = donot take
gini = 0.4082
samples = 7
value = [5, 2]
class = donot take
gini = 0.4444
samples = 3
value = [1, 2]
class = take
gini = 0.0
samples = 2
value = [0, 2]
class = take
gini = 0.0
samples = 2
value = [2, 0]
class = donot take
pagerank less than middle ≤ 0.5
gini = 0.48
samples = 5
value = [3, 2]
class = donot take
data ≤ 0.5
gini = 0.4938
samples = 9
value = [5, 4]
class = donot take
gini = 0.0
samples = 1
value = [1, 0]
class = donot take
gini = 0.4898
samples = 14
value = [8, 6]
class = donot take
gini = 0.0
samples = 1
value = [1, 0]
class = donot take
gini = 0.5
samples = 4
value = [2, 2]
class = donot take
gini = 0.0
samples = 1
value = [0, 1]
class = take
gini = 0.4922
samples = 48
value = [21, 27]
class = take
gini = 0.4938
samples = 9
value = [5, 4]
class = donot take
gini = 0.32
samples = 5
value = [4, 1]
class = donot take
gini = 0.0
samples = 2
value = [0, 2]
class = take
gini = 0.0
samples = 1
value = [1, 0]
class = donot take
gini = 0.5
samples = 4
value = [2, 2]
class = donot take
gini = 0.5
samples = 4
value = [2, 2]
class = donot take
gini = 0.48
samples = 5
value = [3, 2]
class = donot take

Рис. 7 Дерево принятия решений, полученное при обучении на одном тексте

Кроме стандартных алгоритмов построения одного дерева принятия решения также рассматривался случайный лес. Случайный лес (Random Forest) [27] - алгоритм машинного обучения, заключающийся в использовании ансамбля решающих деревьев. Его реализация также имеется в sklearn.

	test1	test2	test3	test4	test5	test6
Дерево, обученное на одном тексте	0.8340	0.6226	0.7843	0.7741	0.8687	0.7611
Дерево, обученное на размеченной выборке	0.9279	0.8644	0.9120	0.9194	0.9594	0.9478
Случайный лес, обученный на размеченной выборке	0.788	0.8319	0.8514	0.8719	0.9182	0.8436

Таблица 6. Значения косинусного коэффициента

	test1	test2	test3	test4	test5	test6
Дерево, обученное на одном тексте	0.1398	0.3288	0.1671	0.1709	0.1643	0.2495
Дерево, обученное на размеченной выборке	0.0802	0.1175	0.0631	0.0629	0.0564	0.0489
Случайный лес, обученный на размеченной выборке	0.1888	0.1223	0.1151	0.1189	0.1202	0.1738

Таблица 7. Значения дивергенции Дженсона-Шеннона

Результаты, полученные для трёх обученных классификаторов представлены в таблицах 6 и 7. Как и раньше для оценки качества используем значения косинусного коэффициента и дивергенции Дженсона-Шеннона, а в качестве тестового набора выступают шесть текстов, принадлежащих различным тематикам.

На основании полученных результатов можно заключить, что наилучший результат показало дерево, обученное на тестовом множестве, а наихудший - дерево обученное на единственном тексте. Кроме того, использование случайного леса не улучшило полученный результат.

Как и для алгоритма TextRank, продемонстрируем результат работы обученных классификаторов на первой публикации тестового набора.

Результат работы дерева, обученного на одном тексте:

Двое из трех пострадавших, которые получили наиболее серьезные травмы при взрыве гранаты в компьютерном клубе в дагестанском селении Агвали, прооперированы. Он отметил, что операции сделали пострадавшим 10 и 20 лет. «Мы ожидаем прибытия из Москвы главных специалистов — детского хирурга и реаниматолога, которые примут решение — проводить этому ребенку операцию на месте или перевести его в Москву», — рассказал Ибрагимов. Третьему пострадавшему также проводится плановая операция. Ее принес 20-летний молодой человек, предположительно, сын хозяина здания. Он рассказал, что нашел ее на улице и решил, что это муляж.

Результат работы дерева, обученного размеченной выборке:

Двое из трех пострадавших, которые получили наиболее серьезные травмы при взрыве гранаты в компьютерном клубе в дагестанском селении Агвали, прооперированы. Он отметил, что операции сделали пострадавшим 10 и 20 лет. Ребенок остается в тяжелом состоянии, осколки попали ему в мозг. «Мы ожидаем прибытия из Москвы главных специалистов — детского хирурга и реаниматолога, которые примут решение — проводить этому ребенку операцию на месте или перевести его в Москву», — рассказал Ибрагимов. Третьему пострадавшему также проводится плановая операция. Ранее стало известно, что троих пострадавших на вертолете Росгвардии доставили в Махачкалу. Ее принес 20-летний молодой человек, предположительно, сын хозяина здания. В результате ЧП погиб школьник, еще 11 (по другим данным, 13) человек получили ранения. Первоначально сообщалось, что взрыв произошел в сельской школе. Он рассказал, что нашел ее на улице и решил, что это муляж.

Результат работы случайного леса:

Двое из трех пострадавших, которые получили наиболее серьезные травмы при взрыве гранаты в компьютерном клубе в дагестанском селении Агвали, прооперированы. Он отметил, что операции сделали пострадавшим 10 и 20 лет. «Мы ожидаем прибытия из Москвы главных специалистов — детского хирурга и реаниматолога, которые примут решение — проводить этому ребенку операцию на месте или перевести его в Москву», — рассказал Ибрагимов. Третьему пострадавшему также проводится плановая операция. 24 апреля в селении Агвали Цумадинского района Дагестана боевая ручная граната взорвалась в здании компьютерного клуба. Ее принес 20-летний молодой человек, предположительно, сын хозяина здания. Он рассказал, что нашел ее на улице и решил, что это муляж.

Первый классификатор показал достойный результат. Тем не менее использовать его на практике не имеет смысла, поскольку он ориентирован на новости определенного вида. И то, что он адекватно справился с рассматриваемым примером, не гарантирует его корректной работы в дальнейшем.

Оба оставшихся классификатора, в отличие от алгоритма TextRank, включают в обзор только значимые предложения. Таким образом, удалось избежать включения нерелевантных предложений в аннотацию. Однако, первый из полученных обзоров наиболее подробно описывает произошедшее событие. Поэтому для решения поставленной задачи будем использовать классификатор, состоящий из одного дерева.

Теперь когда были выбраны методы для решения поставленной задачи, объединим их в одну программу и проверим её работу на изначальной тестовой выборке. Для демонстрации итога работы приведём один из кластеров и составленные обзоры для его элементов:

Первый документ полученного кластера:

Молодежная сборная России обыграла команду Словакии в заключительном матче группового этапа чемпионата мира по хоккею и вышла в четвертьфинал турнира. Об этом сообщает корреспондент «Ленты.ру». Встреча состоялась в ночь на 1 января в Торонто и завершилась победой россиян — 2:0. На 30-й минуте отличился Денис Гурьянов, на 50-й окончательный счет установил Яков Тренин. Голкипер сборной России Илья Самсонов отразил 15 бросков и впервые на турнире сыграл матч на ноль. Сборная России с 6 очками завершила этап на 3-м месте группы B и в четвертьфинале встретится с командой Дании, ставшей 2-й в группе A. Встреча состоится в ночь на 3 января и начнется в 1:30 мск. В случае успеха россияне сыграют в полуфинале с победителем другого четвертьфинала США — Швейцария. В других четвертьфиналах сборная Канады встретится с чехами, а шведы поборются за выход в следующий этап со словаками.

Второй документ полученного кластера:

Появилось видео с лучшими моментами матча молодежного чемпионата мира между Россией и Словакией. Ролик опубликован в Twitter Международной федерации хоккея. Встреча состоялась в ночь на 1 января в «Эйр Канада — Центре» в Торонто и завершилась победой россиян со счетом 2:0. Первую шайбу на 30-й минуте забросил Денис Гурьянов, автором второго точного броска на 50-й минуте стал Яков Тренин. Россияне вышли в четвертьфинал турнира, где в ночь на 3 января сыграют с датчанами.

Полученные обзоры для документов представленного кластера:

Обзор первого документа:

Молодежная сборная России обыграла команду Словакии в заключительном матче группового этапа чемпионата мира по хоккею и вышла в четвертьфинал турнира. Встреча состоялась в ночь на 1 января в Торонто и завершилась победой россиян — 2:0. Сборная России с 6 очками завершила этап на 3-м месте группы B и в четвертьфинале встретится с командой Дании, ставшей 2-й в группе A. Встреча состоится в ночь на 3 января и начнется в 1:30 мск. В случае успеха россияне сыграют в полуфинале с победителем другого четвертьфинала США — Швейцария.

Обзор второго документа:

Появилось видео с лучшими моментами матча молодежного чемпионата мира между Россией и Словакией. Встреча состоялась в ночь на 1 января в «Эйр Канада — Центре» в Торонто и Россияне вышли в четвертьфинал турнира, где в ночь на 3 января сыграют с датчанами.

Оба документа представленного кластера действительно содержат описание одного и того же события, а именно хоккейного матча между Россией и Словакией, а полученные обзоры не содержат лишней информации, что удовлетворяет нашим требованиям.

Выводы

Таким образом, поставленная задача была разбита на две подзадачи: кластеризации рассматриваемой коллекции новостных публикаций и составления обзора событий для полученных кластеров.

Для решения задачи кластеризации были изучены и опробованы графовые и статистические методы. Однако, исследование показало, что алгоритмы основанные на графах (Walktrap и Infomap), не способны должным образом разбить коллекцию русскоязычных новостных документов на группы, отвечающие одному событию. В то время как наиболее популярные статистические алгоритмы k-means и k-means++ справляются с данной задачей. Кроме того, предпринятая попытка улучшения качества разбиения с помощью метода главных компонент (PCA) оказалась успешной, что позволило увеличить значение F-меры на пятнадцать процентов.

Для составления обзора событий был изучен и применён алгоритм автоматического автореферирования TextRank. Однако, данный метод предполагает знание количества требуемых предложений, что в ряде случаев является неприемлемым. Поскольку новостные сообщения могут состоять как из трёх, так и из тридцати предложений, в то время как процент значимых в них предложений может варьироваться. В связи с этим, задача составления обзора событий была сведена к задаче бинарной классификации. Для её решения были изучены методы построения деревьев принятия решений ID3, C4.5 и CART. В результате чего, используя алгоритм CART, был обучен классификатор, принимающий решения по включению рассматриваемого предложения в обзор. Помимо этого был также обучен классификатор, основанный на алгоритме Random

Forest. Однако, он показал несколько худшие результаты по сравнению с предыдущим.

Таким образом, наилучшей комбинацией методов для решения поставленной задачи будем считать алгоритм k-means++, использованный совместно с методом главных компонент, и классификатор, использующий метод CART.

Заключение

В результате выполнения данной работы была разработана программа, которая составляет обзор событий для имеющейся коллекции русскоязычных новостных документов. Кроме этого были размечены тестовая выборка для оценки качества кластеризации и обучающая выборка предложений для классификации, а также рассмотрены популярные алгоритмы машинного обучения. Для достижения поставленной цели дополнительно был изучен язык программирования Python и некоторые его библиотеки.

В дальнейшем планируется составлять обзор событий для коллекции новостных публикаций, полученной с различных новостных порталов, что подразумевает дублирование ряда новостей.

С исходным кодом разработанной программы можно ознакомиться по ссылке https://github.com/anastasia2145/vkr.git

Литература

1) Мандель И. Д. Кластерный анализ. — М.: Финансы и Статистика, 1988
2) Воронцов К.В. Методы кластеризации: курс лекций. Режим доступа: http://www.machinelearning.ru/wiki/(дата обращения 31.03.17)
3) Segaran T. Programming Collective Intelligence. Sebastool: O'RELLY, 2008. 368 p.
4) Van Rijsbergen, C. J., 'Information Retrieval', London, 1979
5) Киселев М. В. Пивоваров В. С. Шмулевич М. М. Метод кластеризации текстов, учитывающий совместную встречаемость ключевых терминов, и его применение к анализу тематической структуры новостного потока, а также ее динамики.
6) Stuart P. Lloyd Least Squares Quantization in PCM
7) Arthur D., Vassilvitskii S. K-means++: the advantages of careful seeding / SODA'07 Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms. — CityPhiladelphia, StatePA: placecountry-regionSIAM Press. — 2007. — P. 1027–1035.
8) Pons P., Latapy M. Computing communities in large networks using random walks // Computer and Information Sciences-ISCIS. 2005. P. 284–293.
9) Joe H. Ward Hierarchical grouping to optimize an objective function // Journal of the American statistical association, 58(301):236–244, 1963.
10) Rosvall M., Axelsson D., Bergstrom C. T. The map equation // The European Physical Journal Special Topics. 2009. Vol. 178, No 1. P. 13–23.
11) Левитин А. В. Жажные методы: Алгоритм Хаффмана //

Алгоритмы. Введение в разработку и анализ. М.: Вильямс, 2006 С. 392-398

12) Tipping M., Bishop C. Probabilistic Principal Component Analysis // Journal of the Royal Statistical Society, Series B, 61, Part 3, P. 611-622

13) Гантмахер Ф. Р. Теория матриц. — М.: Наука, 1966. С. 576

14) Mihalcea R., Tarau P. TextRank: Bringing Order into Texts, 2004

15) Усталов Д. А. Извлечение терминов из русскоязычных текстов при помощи графовых моделей, 2012

16) *Паклин Н.Б., Орешков В.И.* Глава 9. // Бизнес-аналитика: от данных к знаниям(+CD): Учебное пособие. 2-е изд.. — СПб: Питер, 2013. — С. 444-459.

17) Breiman L., Friedman J. H., Olshen R. A., & Stone C. J. Classification and regression trees. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software, 1984

18) Новостной портал Lenta [Электронный ресурс]: URL:http://lenta.ru

19) Korobov M.: Morphological Analyzer and Generator for Russian and
Ukrainian Languages // Analysis of Images, Social Networks and Texts, 2015. P. 320-332

20) Porter, Martin F. An Algorithm for Suffix Stripping

21) Snowball stemmer [Электронный ресурс]: URL:http://snowball.tartarus.org/algorithms/russian/stemmer.html

22) *Jones K. S.* A statistical interpretation of term specificity and its application in retrieval // Journal of Documentation // MCB University: MCB University Press, 2004. — Т. 60, No 5. С. 493-502.

23) Le Martelot E., Hankin C. Fast multi-scale detection of relevant communities in large-scale networks // The Computer Journal 2013. Vol.

56, No 9. P. 1136–1150.

24) David M W Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation // Journal of Machine Learning Technologies, 2011

25) Louis A., Nenkova A. Automatic Summary Evaluation without Human Models // University of Pennsylvania Philadelphia

26) Brin S., Page L. The Anatomy of a Large-Scale Hypertextual Web Search

27) Breiman L., Cutler A. Random Forests