Министерство образования и науки Российской Федерации

Калужский филиал
федерального государственного бюджетного образовательного
учреждения высшего образования
**«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»**
(КФ МГТУ им. Н.Э. Баумана)

Ю.С. Белов, С.С. Гришунов

MAHOUT. АЛГОРИТМЫ КЛАСТЕРИЗАЦИИ

Методические указания по выполнению лабораторной работы
по курсу «Технологии обработки больших данных»

Калуга - 2018

УДК 004.62
ББК 32.972.5
Б435

Методические указания составлены в соответствии с учебным планом КФ МГТУ им. Н.Э. Баумана по направлению подготовки 09.03.04 «Программная инженерия» кафедры «Программного обеспечения ЭВМ, информационных технологий и прикладной математики».

Методические указания рассмотрены и одобрены:

- Кафедрой «Программного обеспечения ЭВМ, информационных технологий и прикладной математики» (ФН1-КФ) протокол №_6_ от «_12_» января 2018 г.

Зав. кафедрой ФН1-КФ _____ д.ф.-м.н., профессор Б.М. Логинов

- Методической комиссией факультета ФНК протокол № 1 от «30» января 2018 г.

Председатель методической
комиссии факультета ФНК _____ к.х.н., доцент К.Л. Анфилов

- Методической комиссией
КФ МГТУ им.Н.Э. Баумана протокол № 1 от «06» 02 2018 г.

Председатель методической комиссии
КФ МГТУ им.Н.Э. Баумана
_____ д.э.н., профессор О.Л. Перерва

Рецензент:
к.т.н., зав. кафедрой ЭИУ2-КФ _____ И.В. Чухраев

Авторы
к.ф.-м.н., доцент кафедры ФН1-КФ _____ Ю.С. Белов
ассистент кафедры ФН1-КФ _____ С.С. Гришунов

Аннотация

Методические указания по выполнению лабораторной работы по курсу «Технологии обработки больших данных» содержат краткое описание наиболее используемых метрик расстояния между объектами и алгоритмов кластеризации. Рассмотрены примеры реализации кластеризации больших данных различными методами с помощью библиотеки Mahout.

Предназначены для студентов 4-го курса бакалавриата КФ МГТУ им. Н.Э. Баумана, обучающихся по направлению подготовки 09.03.04 «Программная инженерия».

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Настоящие методические указания составлены в соответствии с программой проведения лабораторных работ по курсу «Технологии обработки больших данных» на кафедре «Программное обеспечение ЭВМ, информационные технологии и прикладная математика» факультета фундаментальных наук Калужского филиала МГТУ им. Н.Э. Баумана.

Методические указания, ориентированные на студентов 4-го курса направления подготовки 09.03.04 «Программная инженерия», содержат краткое описание алгоритмов кластеризации, примеры их реализации с помощью библиотеки Mahout и задание на выполнение лабораторной работы.

Методические указания составлены для ознакомления студентов с библиотекой Mahout. Для выполнения лабораторной работы студенту необходимы минимальные знания по программированию на высокоуровневом языке программирования Java.

ЦЕЛЬ И ЗАДАЧИ РАБОТЫ, ТРЕБОВАНИЯ К РЕЗУЛЬТАТАМ ЕЕ ВЫПОЛНЕНИЯ

Целью выполнения лабораторной работы является формирование практических навыков работы с библиотекой Mahout для кластеризации больших данных.

Основными задачами выполнения лабораторной работы являются:

1. Получить навыки работы с Apache Mahout.
2. Изучить алгоритмы кластеризации.
3. Научиться реализовывать кластеризацию данных с помощью Apache Mahout
4. Получить навыки векторизации текстовых документов с помощью Apace Mahout.

Результатами работы являются:

- Входные файлы с данными для обучения системы
- Программа, реализующая рекомендательную систему
- Результаты тестирования программы
- Подготовленный отчет

КРАТКАЯ ХАРАКТЕРИСТИКА ОБЪЕКТА ИЗУЧЕНИЯ, ИССЛЕДОВАНИЯ

Apache Mahout – это библиотека для работы с алгоритмами машинного обучения, которая может быть использована как надстройка к Hadoop или самостоятельно. В библиотеке реализованы методы коллаборативной фильтрации, кластеризации и классификации.

Кластеризация (или кластерный анализ) — это задача разбиения множества объектов на группы, называемые кластерами. Внутри каждой группы должны оказаться «похожие» объекты, а объекты разных группы должны быть как можно более отличны. Главное отличие кластеризации от классификации состоит в том, что перечень групп четко не задан и определяется в процессе работы алгоритма.

Применение кластерного анализа в общем виде сводится к следующим этапам:

1. Отбор выборки объектов для кластеризации.
2. Определение множества переменных, по которым будут оцениваться объекты в выборке. При необходимости – нормализация значений переменных.
3. Вычисление значений меры сходства между объектами.
4. Применение метода кластерного анализа для создания групп сходных объектов (кластеров).
5. Представление результатов анализа.

После получения и анализа результатов возможна корректировка выбранной метрики и метода кластеризации до получения оптимального результата.

Как же определять «похожесть» объектов? Для начала нужно составить вектор характеристик для каждого объекта — как правило, это набор числовых значений, например, рост-вес человека. Однако существуют также алгоритмы, работающие с качественными (т.н. категорийными) характеристиками.

После того, как был определилен вектор характеристик, можно провести нормализацию, чтобы все компоненты давали одинаковый

вклад при расчете «расстояния». В процессе нормализации все значения приводятся к некоторому диапазону, например, [-1, -1] или [0, 1].

Наконец, для каждой пары объектов измеряется «расстояние» между ними — степень похожести. Существует множество метрик, основными из которых являются:

Евклидово расстояние — наиболее распространенное расстояние. Оно является геометрическим расстоянием в многомерном пространстве.

Квадрат евклидова расстояния. Иногда может возникнуть желание возвести в квадрат стандартное евклидово расстояние, чтобы придать большие веса более отдаленным друг от друга объектам.

Расстояние городских кварталов (манхэттенское расстояние). Это расстояние является просто средним разностей по координатам. В большинстве случаев эта мера расстояния приводит к таким же результатам, как и для обычного расстояния Евклида. Однако отметим, что для этой меры влияние отдельных больших разностей (выбросов) уменьшается (так как они не возводятся в квадрат).

Расстояние Чебышева. Это расстояние может оказаться полезным, когда желают определить два объекта как «различные», если они различаются по какой-либо одной координате (каким-либо одним измерением).

Степенное расстояние. Иногда желают прогрессивно увеличить или уменьшить вес, относящийся к размерности, для которой соответствующие объекты сильно отличаются. Это может быть достигнуто с использованием степенного расстояния.

Выбор расстояния (критерия схожести) лежит полностью на исследователе. При выборе различных мер результаты кластеризации могут существенно отличаться.

Рассмотрим несколько наиболее распространенных алгоритмов кластеризации.

Алгоритм k-means (k-средних)

Наиболее простой, но в то же время достаточно неточный метод кластеризации в классической реализации. Он разбивает множество элементов векторного пространства на заранее известное число кластеров k. Действие алгоритма таково, что он стремится минимизировать среднеквадратичное отклонение на точках каждого кластера. Основная идея заключается в том, что на каждой итерации перевычисляется центр масс для каждого кластера, полученного на предыдущем шаге, затем векторы разбиваются на кластеры вновь в соответствии с тем, какой из новых центров оказался ближе по выбранной метрике. Алгоритм завершается, когда на какой-то итерации не происходит изменения кластеров.

Алгоритм c-means (c-средних)

Вместо однозначного ответа на вопрос к какому кластеру относится объект, данный алгоритм определяет вероятность того, что объект принадлежит к тому или иному кластеру, также при перерасчете центра масс каждый объект участвует только в той доле, в которой он принадлежит кластеру. Таким образом, утверждение «объект А принадлежит к кластеру 1 с вероятностью 90%, к кластеру 2 — 10% верно и более удобно.

Алгоритм выделения связных компонент

В алгоритме выделения связных компонент задается входной параметр *R* и в графе удаляются все ребра, для которых «расстояния» больше *R*. Соединенными остаются только наиболее близкие пары объектов. Смысл алгоритма заключается в том, чтобы подобрать такое значение *R*, лежащее в диапазон всех «расстояний», при котором граф «развалится» на несколько связных компонент. Полученные компоненты и есть кластеры.

Для подбора параметра *R* обычно строится гистограмма распределений попарных расстояний. В задачах с хорошо выраженной кластерной структурой данных на гистограмме будет два пика – один соответствует внутрикластерным расстояниям, второй –

межкластерным расстояния. Параметр R подбирается из зоны минимума между этими пиками. При этом управлять количеством кластеров при помощи порога расстояния довольно затруднительно.

Алгоритм минимального покрывающего дерева

Алгоритм минимального покрывающего дерева сначала строит на графе минимальное покрывающее дерево, а затем последовательно удаляет ребра с наибольшим весом. Таким образом на графе остается несколько несвязанных остовных деревьев, каждое из которых определяет отдельный кластер.

Также существует множество модификаций данный алгоритмов, алгоритмы, основанные на статистических моделях, на искусственных нейронных сетях и др.

ПРИМЕР ПОСТРОЕНИЯ АЛГОРИТМА КЛАСТЕРИЗАЦИИ

Рассмотрим встроенные средства Apache Mahout для решения задачи кластеризациии.

Рассмотрим пример построения алгоритма кластеризации K-means.

```
public static final double[][] points = { {1, 1}, {2, 1}, {1, 2},
    {2, 2}, {3, 3}, {8, 8}, {9, 8}, {8, 9}, {9, 9}};

public static void writePointsToFile(List<Vector> points,
        String fileName,
        FileSystem fs,
        Configuration conf) throws IOException {
          Path path = new Path(fileName);
          SequenceFile.Writer writer = new SequenceFile.Writer(fs, conf,
              path, LongWritable.class, VectorWritable.class);
          long recNum = 0;
          VectorWritable vec = new VectorWritable();
          for (Vector point : points) {
              vec.set(point);
              writer.append(new LongWritable(recNum++), vec);
          }
          writer.close();
}
public static List<Vector> getPoints(double[][] raw) {
        List<Vector> points = new ArrayList<Vector>();
        for (int i = 0; i < raw.length; i++) {
                  double[] fr = raw[i];
                  Vector vec = new RandomAccessSparseVector(fr.length);
                  vec.assign(fr);
                  points.add(vec);
        }
        return points;
}
```

```
public static void main(String args[]) throws Exception {
    int k = 2;
    List<Vector> vectors = getPoints(points);
    File testData = new File("testdata");
    if (!testData.exists()) {
        testData.mkdir();
    }
    testData = new File("testdata/points");
    if (!testData.exists()) {
        testData.mkdir();
    }
    Configuration conf = new Configuration();
    FileSystem fs = FileSystem.get(conf);
    writePointsToFile(vectors, "testdata/points/file1", fs, conf);
    Path path = new Path("testdata/clusters/part-00000");
    SequenceFile.Writer writer = new SequenceFile.Writer(
        fs, conf, path, Text.class, Cluster.class);
    for (int i = 0; i < k; i++) {
        Vector vec = vectors.get(i);
        Cluster cluster = new Cluster(
            vec, i, new EuclideanDistanceMeasure());
        writer.append(new Text(cluster.getIdentifier()), cluster);
    }
    writer.close();
    KMeansDriver.run(conf, new Path("testdata/points"),
        new Path("testdata/clusters"),
        new Path("output"), new EuclideanDistanceMeasure(),
        0.001, 10, true, false);
    SequenceFile.Reader reader = new SequenceFile.Reader(fs,
        new Path("output/" + Cluster.CLUSTERED_POINTS_DIR
        + "/part-m-00000"), conf);
    IntWritable key = new IntWritable();
    WeightedVectorWritable value = new WeightedVectorWritable();
```

```
        while (reader.next(key, value)) {
                System.out.println(
                value.toString() + " belongs to cluster "
                        + key.toString());
        }
        reader.close();
    }
```

Создадим тестовый набор точек. Так как алгоритмы Mahout работают с объектами Vector, то преобразуем массив точек в массив векторов:

```
List<Vector> vectors = getPoints(points);
```

Алгоритмы кластеризации (табл. 1) Mahout могут выполняться в двух режимах: непосредственно в памяти и с помощью MapReduce фреймворка. Для обработки больших объемов данных реализации алгоритмов, работающие в памяти, не могут быть использованы, так как им не будет хватать оперативной памяти для размещения всех элементов. Поэтому в таком случае используется реализация на основе MapReduce подхода. MapReduce работает с файлами, поэтому входные данные должны быть также представлены в виде файла.

Таблица 1

Алгоритмы кластеризации, реализованные в Mahout

Алгоритм	**Реализация в памяти**	**Реализация MapReduce**
K-means	KMeansCluster	KMeansDriver
Canopy	CanopyCluster	CanopyDriver
Fuzzy k-means	FuzzyKMeansCluster	FuzzyKMeansDriver
Dirichlet	DirichletCluster	DirichletDriver
LDA	–	LDADriver

Создадим файл и запишем в него тестовые данные:

```
File testData = new File("testdata");
if (!testData.exists()) {
        testData.mkdir();
}
testData = new File("testdata/points");
if (!testData.exists()) {
        testData.mkdir();
}
Configuration conf = new Configuration();
FileSystem fs = FileSystem.get(conf);
writePointsToFile(vectors,
        "testdata/points/file1", fs, conf);
```

Начальные значения могут быть созданы пользователем по какой-то логике, либо можно задать случайные значения с помощью RandomSeedGenerator. Необходимо создать объект Cluster, его конструктор принимает в качестве параметров: центр кластера, номер кластера и метрику расстояния.

```
SequenceFile.Writer writer = new SequenceFile.Writer(
    fs, conf, path, Text.class, Cluster.class);
 for (int i = 0; i < k; i++) {
    Vector vec = vectors.get(i);
    Cluster cluster = new Cluster(
    vec, i, new EuclideanDistanceMeasure());
    writer.append(new Text(cluster.getIdentifier()), cluster);
}
```

Запускаем алгоритм кластеризации K-means. Для запуска MapReduce задачи необходимо передать в метод run следующие параметры:

1. Конфигурацию HDFS, для операций с файлами
2. Файл, содержащий входные векторы

3. Файл, содержащий начальную конфигурацию кластеров (начальные положения центров)

4. Метрика расстояния. Метрика, вычисляющая расстояние между двумя векторами (табл. 2).

5. Порог сходимости. Если во время итераций центры кластера не будут перемещаться на расстояние большее, чем на данное значение, то выполнение алгоритма будет остановлено

6. Максимальное количество итераций. Если порог сходимости не будет достигнут, то выполнение алгоритма остановится через данное количество итераций.

Таблица 2

Метрики расстояния, реализованные в Mahout

Реализация	**Вычисление расстояния**
EuclideanDistance Measure	$d = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + ... + (a_n - b_n)^2}$
SquaredEuclidean DistanceMeasure	$d = (a_1 - b_1)^2 + (a_2 - b_2)^2 + ... + (a_n - b_n)^2$
ManhattanDistance Measure	$d = \lvert a_1 - b_1 \rvert + \lvert a_2 - b_2 \rvert + ... + \lvert a_n - b_n \rvert$
CosineDistance Measure	$d = 1 - \frac{(a_1 b_1 + a_2 b_2 + ... + a_n b_n)}{\sqrt{(a_1^2 + a_2^2 + ... + a_n^2)}\sqrt{(b_1^2 + b_2^2 + ... + b_n^2)}}$
TanimotoDistance Measure	$d = 1 - \frac{(a_1 b_1 + a_2 b_2 + ... + a_n b_n)}{\sqrt{(a_1^2 + a_2^2 + ... + a_n^2)} + \sqrt{(b_1^2 + b_2^2 + ... + b_n^2)} - (a_1 b_1 + a_2 b_2 + ... + a_n b_n)}$

```
KMeansDriver.run(conf, new Path("testdata/points"),
        new Path("testdata/clusters"),
        new Path("output"), new EuclideanDistanceMeasure(),
        0.001, 10, true, false);
```

Считываем результаты работы алгоритма и выводим на экран каждую точку и номер кластера, которому она принадлежит:

```
SequenceFile.Reader reader = new SequenceFile.Reader(fs,
        new Path("output/" + Cluster.CLUSTERED_POINTS_DIR
        + "/part-m-00000"), conf);
  IntWritable key = new IntWritable();
  WeightedVectorWritable value = new WeightedVectorWritable();
  while (reader.next(key, value)) {
        System.out.println(
        value.toString() + " belongs to cluster "
                + key.toString());
  }
  reader.close();
```

Также Mahout имеет встроенные средства для представления текстовых файлов в виде векторов на основе метрики TF-IDF: Analyzer, Document Processor, DictionaryVectorizer.

ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ

Задание 1.

Изучить средства Mahout для векторизации текстов. Реализовать кластеризацию статей по темам. Исходные статьи можно загрузить из коллекции Reuters: http://www.daviddlewis.com/resources/testcollections/

Можно использовать любой алгоритм кластеризации и любую метрику.

Результат сохранить в виде:

Номер кластера – Список статей.

Подобрать названия для созданных кластеров (темы статей).

Задание 2

Создать тестовый файл с координатами точек на плоскости. Реализовать алгоритм кластеризации точек (согласно варианту) с различными метриками. Результаты сохранить в файл, затем графически отобразить полученные кластеры с помощью любого средства. Сравнить результаты.

ТРЕБОВАНИЯ К РЕАЛИЗАЦИИ

Приложение должно быть реализовано на языке java. В качестве источника данных для кластеризации можно пользоваться как файлами, размещенными в HDFS, так и файлами в локальной файловой системе.

ВАРИАНТЫ ЗАДАНИЙ

1. Алгоритм: K-means
   Метрики: SquaredEuclideanDistanceMeasure, CosineDistanceMeasure
2. Алгоритм: K-means
   Метрики: TanimotoDistanceMeasure, ManhattanDistanceMeasure
3. Алгоритм: K-means
   Метрики: TanimotoDistanceMeasure, CosineDistanceMeasure

4. Алгоритм: Canopy
   Метрики: EuclideanDistanceMeasure, CosineDistanceMeasure
5. Алгоритм: Canopy
   Метрики: SquaredEuclideanDistanceMeasure, ManhattanDistanceMeasure
6. Алгоритм: Dirichlet
   Метрики: TanimotoDistanceMeasure, CosineDistanceMeasure
7. Алгоритм: Dirichlet
   Метрики: ManhattanDistanceMeasure, CosineDistanceMeasure
8. Алгоритм: Fuzzy k-means
   Метрики: SquaredEuclideanDistanceMeasure, ManhattanDistanceMeasure
9. Алгоритм: Fuzzy k-means
   Метрики: SquaredEuclideanDistanceMeasure, TanimotoDistanceMeasure
10. Алгоритм: LDA
    Метрики: EuclideanDistanceMeasure, CosineDistanceMeasure

КОНТРОЛЬНЫЕ ВОПРОСЫ И ЗАДАНИЯ

1. Раскройте область применения библиотеки Apache Mahout.
2. Дайте определение кластеризации.
3. Приведите основные этапы кластеризации.
4. Перечислите основные метрики определения подобия объектов.
5. Раскройте сущность алгоритма k-means.
6. Раскройте сущность алгоритма c-means.
7. Раскройте сущность алгоритма выделения связных компонент.
8. Раскройте сущность алгоритма минимального покрывающего дерева.
9. Приведите команды для запуска процесса кластеризации каким-либо алгоритмом, используя библиотеку Mahout.
10. Приведите параметры, которые необходимо передать для запуска MapReduce задачи для кластеризации данных.

ФОРМА ОТЧЕТА ПО ЛАБОРАТОРНОЙ РАБОТЕ

На выполнение лабораторной работы отводится 3 занятия (6 академических часов: 5 часов на выполнение и сдачу лабораторной работы и 1 час на подготовку отчета).

Номер варианта студенту выдается преподавателем.

Отчет на защиту предоставляется в печатном виде.

Структура отчета (на отдельном листе(-ах)): титульный лист, формулировка задания (вариант), этапы выполнения работы (со скриншотами), результаты выполнения работы. выводы.

ОСНОВНАЯ ЛИТЕРАТУРА

1. Федин Ф.О. Анализ данных. Часть 1. Подготовка данных к анализу [Электронный ресурс] : учебное пособие / Ф.О. Федин, Ф.Ф. Федин. — Электрон. текстовые данные. — М. : Московский городской педагогический университет, 2012. — 204 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/26444.html
2. Федин Ф.О. Анализ данных. Часть 2. Инструменты Data Mining [Электронный ресурс] : учебное пособие / Ф.О. Федин, Ф.Ф. Федин. — Электрон. текстовые данные. — М. : Московский городской педагогический университет, 2012. — 308 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/26445.html
3. Чубукова, И.А. Data Mining [Электронный ресурс] : учеб. пособие — Электрон. дан. — Москва : , 2016. — 470 с. — Режим доступа: https://e.lanbook.com/book/100582. — Загл. с экрана.
4. Воронова Л.И. Big Data. Методы и средства анализа [Электронный ресурс] : учебное пособие / Л.И. Воронова, В.И. Воронов. — Электрон. текстовые данные. — М. : Московский технический университет связи и информатики, 2016. — 33 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/61463.html

ДОПОЛНИТЕЛЬНАЯ ЛИТЕРАТУРА

5. Волкова Т.В. Разработка систем распределенной обработки данных [Электронный ресурс] : учебно-методическое пособие / Т.В. Волкова, Л.Ф. Насейкина. — Электрон. текстовые данные. — Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2012. — 330 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/30127.html
6. Кухаренко Б.Г. Интеллектуальные системы и технологии [Электронный ресурс] : учебное пособие / Б.Г. Кухаренко. — Электрон. текстовые данные. — М. : Московская государственная академия водного транспорта, 2015. — 116 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/47933.html

7. Воронова Л.И. Интеллектуальные базы данных [Электронный ресурс] : учебное пособие / Л.И. Воронова. — Электрон. текстовые данные. — М. : Московский технический университет связи и информатики, 2013. — 35 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/63324.html
8. Николаев Е.И. Базы данных в высокопроизводительных информационных системах [Электронный ресурс] : учебное пособие / Е.И. Николаев. — Электрон. текстовые данные. — Ставрополь: Северо-Кавказский федеральный университет, 2016. — 163 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/69375.html

Электронные ресурсы:

9. http://hadoop.apache.org/ (англ.)
10. http://mahout.apache.org/ (англ.)