Шпаргалка
Домашняя работа по теме: «Сад камней»

Домашняя работа состоит из 2 заданий, который нужно будет выполнить последовательно. Задание #2 не будет проверяться в случае, если не сдано задание #1. Во избежание неприятных ситуаций следует решения обоих заданий высылать с одной и той же почты.

Адреса:

- $help\_infosearch@mail.ru$ — все вопросы, уточнения, просьбы и жалобы шлите сюда
- $sekitei1\_infosearch@mail.ru$ — первое ДЗ (извлечение фич)
- $sekitei2\_infosearch@mail.ru$ — второе ДЗ (кластеризация)

1. Задание 1

1.1. Цель

Выделить признаки из урлов через алгоритм "Сад камней".

1.2. Данные

В папке $./data/$ лежат ссылки:

- $*.examined$ — QLink'и
- $*.general$ — обычные урлы

Ссылки собраны с 3 сайтов. Это открытый набор данных, на которых вы можете тестировать работоспособность своей программы. Еще 2 сайта — закрытый набор данных, которые будут использоваться для финальной оценки работы.

В папку $./check/$ будет помещаться вывод программы. Там уже есть результаты для 3 сайтов из открытого набора.

Результат работы теста может быть PASSED, что означает тест пройден, или NOT PASSED, что означает тест не пройден и тогда будет показана причина провала теста и имя теста.

Часть кода написана, нужно реализовать метод $extract\_features$ в модуле $extract\_features.py$.

1.3. Запуск

Заменить файл $extract\_features.py$ на свою реализацию и запустить скрипт проверки

$$python \ \ ./check - features.py$$

1.3.1. Параметры запуска

1. Файл $*.examined$
2. Файл $*.general$
3. Файл, в который нужно записать результаты теста

1.3.2. Результаты

Формат файла результатов:

Признак\tКоличество\n
Признак\tКоличество\n
...

Файл результатов должен быть отсортирован по количеству признаков.

1.3.3. Именование фичей

Для сегментов:
$segment\_ < name > \_ < index > :< val >$, где: $name$ — название фичи для сегмента, $index$ — индекс сегмента, val — значение фичи.
Для параметров:
$param\_name :< названиепараметра >$
$param :< ключ = значение >$
Для описанных правил имеем получаем следующие имена фичей

1. Количество сегментов в пути
   $segments :< len >$
2. Список имен параметров запросной части (может быть пустым)
   $param\_name :< имя >$
3. Присутствие в запросной части пары $< parameters = value >$
   $param :< parameters = value >$

4. Сегмент пути на позиции:

   (a) Совпадает со значением $<$ *строка* $>$
   $segment\_name\_ < index >:< string >$

   (b) Состоит из цифр
   $segment\_[0-9]\_ < index >: 1$

   (c) $<$ *строка с точностью до комбинации цифр* $>$
   $segment\_substr[0-9]\_ < index >: 1$

   (d) Имеет заданное расширение
   $segment\_ext\_ < index >:< extension\ value >$

   (e) Комбинация из двух последних вариантов
   $segment\_ext\_substr[0-9]\_ < index >:< extension\ value >$

   (f) Состоит из данного количества символов:
   $segment\_len\_ < index >:< segment\ length >$

   Знак $<>$ означает подстановку значения, например, $< index >$ означает, что нужно использовать индекс сегмента:
   $segment\_substr[0-9]\_1 : 1$ — первый сегмент имеет фичу $substr[0-9]$

1.4. Результаты

Отлаженный скрипт прислать в качестве выполненного ДЗ.

2. Задание 2

2.1. Цель

Разработать алгоритм, имитирующий приоритезацию спайдера на основе алгоритма "Сад камней" с качеством, превышающим "жадный" алгоритм.

2.2. Описание

Используя алгоритм "Сад камней" для извлечения признаков из урлов и любой из алгоритмов кластеризации, определить, нужно ли качать входящий урл или нет. Датасет разделен на два множества: тренировочное (три сайта) и валидационное (два сайта). Для каждого сайта нужно будет максимально эффективно выбрать доступную квоту.

Квота — максимальное количество урлов, которое может быть взято с данного сайта. Это число передается в качестве параметра на вход алгоритму для принятия решения о важности урла.

2.3. Реализация

Студенту нужно реализовать две функции

1. $define\_segments$, выделяющая сегменты из сайта и квоты для них. На вход получает 500 урлов с кулинками, и 500 урлов без кулинок, а также значение квоты для всего сайта в целом.

2. $fetch\_url$, определяющая нужность урла. На вход принимает урл, как параметр, на выход должна вернуть $True$, если урл нужно положить в индекс, иначе — $False$.

2.4. Как все работает

Для некоторой рандомной выборки урлов для инициализации (по 500 урлов для каждого класса) вызывается метод для выделения сегментов сайта ($define\_segments$). Для оставшихся урлов вызывается метод, определяющий ценность урла (если урл нужно положить в индекс, функция должна вернуть $True$, в противном случае — $False$). Если функция $fetch\_url$ возвращает $True$, то тест уменьшает значение квоты — урл "скачан". Обработка заканчивается, когда достигнута граница квоты или больше нет урлов для выборки.

2.5. Метрики

В качестве метрики используется $F1$ мера

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

где $precision$ — полнота выборки, $recall$ — точность выборки.

$$precision = \frac{1}{T} \sum_{i=0}^{T} \frac{N_{fetched}}{N_{quota}}$$

где $N_{fetched}$ — количество выбранных документов, N_{quota} — количество документов, разрешенных квотой, T — количество тестов (сайтов)

$$recall = \frac{1}{T}\sum_{i}^{T}\frac{N_{qfetched}}{N_{qtotal}}$$

где $N_{qfetched}$ — количество отобранных документов с кулинками, N_{qtotal} — количество документов с кулинками всего.

Значение F меры	Баллы
0.7	2
0.8	5
0.9	10

На каждый сайт должно уходить не более 15 секунд.

2.6. Прототипы функций

- $define\_segments(QLINK\_URLS, UNKNOWN\_URLS, QUOTA)$,
  $QLINK\_URLS$ — массив урлов с кулинками,
  $UNKNOWN\_URLS$ — массив урлов без кулинок,
  $QUOTA$ — размер квоты для сайтов.

- $fetch\_url(url)$,
  url — урл для оценки.