자료구조

리뷰 관리 프로그램 만들기

20203063 강수민

데이터 처리 과정 (1)

1. .tsv 파일에서 데이터 추출

```
def load_data(dataset: CircularDoublyLinkedList):
    with open('amazon_reviews_us_Shoes_v1_00_40k.tsv', 'r', encoding='utf-8') as file:
        next(file)
        for line in file:
            fields = line.split('\t')
            for i in range(NUM_OF_CATEGORY):
                dataset.append(fields[i], i)
```

데이터를 일관성 있게 만들기 위해 첫 줄은 건너뛰고 입력 받음.

2. 데이터 필터링 후 그룹화

함수 인자 설명
- filter_str : 필터 조건을 담은 문자열
- group_by : 그룹핑하고 싶은 카테고리를 담은 문자열
- top : 상위 몇 개까지 가져올 것인지

```
def get_data(dataset: CircularDoublyLinkedList, filter_str, group_by, top):
    res_dict = {}
    filtered = dataset.filter(CircularDoublyLinkedListFilter(filter_str))
    filtered = sorted(filtered, key=(itemgetter(CATEGORY_NUM[group_by])))
    group_data = groupby(filtered, key=itemgetter(CATEGORY_NUM[group_by]))

    for key, group_data in group_data:
        res_dict[key] = list(group_data)

    group_sorted = sorted(res_dict.items(), key=lambda x: len(x[1]), reverse=True)
```

itemgetter() : sorted 함수의 key 매개변수에 적용하여 다양한 기준으로 정렬할 수 있도록 함.

groupby() : 반복 가능한 객체를 키값으로 분류하고 그 결과를 반환하는 함수.

정렬을 해야 하는 이유..? => groupby 함수의 특징

데이터 처리 과정 (2)

3. 그룹 별 유용한 투표를 기준으로 내림차순 정렬

```
for ind in range(top):
    group_name = group_sorted[ind][0]
    product_group = group_sorted[ind][1]

    if len(product_group) < top:
        print(f"number of elements is least than {top}")
        return

    product_group.sort(key=lambda x: int(x[CATEGORY_NUM['helpful_votes']]), reverse=True)
    print(f"-------- {group_by} : {group_name} --------\n")
```

-> 정렬부분

4. 사용자에게 출력

```
for j in range(top):
    review_title = product_group[j][CATEGORY_NUM['review_headline']]
    review_body = product_group[j][CATEGORY_NUM['review_body']]
    helpful = product_group[j][CATEGORY_NUM['helpful_votes']]
    print(f"<{j+1}th Review : {review_title}>\n* {review_body}\n- helpful_votes : {helpful}\n")
```

BidirectNode 확장 & Filter, Find 메소드 함수 구현

BidirecNode 구현

```python
class BidirectNode:
    def __init__(self, value, prevNode: 'BidirectNode', nextNode:'BidirectNode', key = -1):
        self.value = value
        self.key = key
        self.prev = prevNode
        self.next = nextNode
```

Filter 함수

```python
def filter(self, haveFindClass):
    typeArgu = type(haveFindClass)
    if 'find' not in dir(haveFindClass):
        print(f"{typeArgu}:{typeArgu.__name__} has no 'find' function.")
        return
    return haveFindClass.find(self)
```

Filter Class & find 함수

라인 별 초기화 조건 ->

조건 순회 & 필터 여부 판별->

```python
class CircularDoublyLinkedListFilter:
    def __init__(self, *conditions : str):
        self.result = list()
        self.conditions = conditions

    def find(self, dataset : 'CircularDoublyLinkedList') -> list:
        count_node = 0
        line_list = list()
        filtered = True
        for key, value in dataset:
            count_node += 1
            line_list.append(value)
            if count_node % NUM_OF_CATEGORY == 0 and line_list:
                if not filtered:
                    self.result.append(line_list)
                    filtered = True
                line_list = list()

                for con_str in self.conditions:
                    left, operator, right = con_str.split(' ')
                    if CATEGORY_NUM[left] == key:
                        if 'A' < value < 'z':
                            if value == right:
                                filtered = False
                        elif eval(''.join([value, operator, right])):
                            filtered = False

        return self.result
```

*실행 예시

```
if __name__ == '__main__':
    dataset = CircularDoublyLinkedList()
    load_data(dataset)
    get_data(dataset, "verified_purchase == Y", "product_title", 3)
    #get_data(dataset, "helpful_votes > 40", "product_id", 2)
```

감사합니다.