# 가설 1 ( 판매량 )

| 작성일시 | @2021년 7월 5일 오후 7:47 |
| --- | --- |

## ♉ 목차

✔️ 5-3) 시각화
✔️ 분석 결과
(6) 상품 카테고리에 속하는 상품 갯수 대비 주문량 비교
✔️ 6-1) 사용 데이터 확인하기
✔️ 6-2) 2개의 DataFrame을 Merge
✔️ 6-3) Department_product DataFrame의 결측치 확인
✔️ 6-4) 상품 갯수 대비 주문량에 대한 파생변수 생성 & 오름차순 정렬
✔️ 6-5) 시각화
✔️ 분석 결과

# ♉ 사용 DataFrame

## (1) OPP

- orders.csv 파일의 eval_set 칼럼값이 prior 인 주문내역에 대한 데이터
- 어떤 상품을 구매했고, 상품을 장바구니에 담은 순서, 재구매한 상품인지에 대한 정보
- 32434489 rows , 4 columns 으로 구성

| Aa column 명 | 기능 설명 | type | NaN값 여부 |
|---|---|---|---|
| order_id | 주문 ID | int64 | X |
| product_id | 주문한 상품ID | int64 | X |
| add_to_cart_order | 장바구니에 담은 순서 | int64 | X |
| reordered | 재구매한 상품인지 아닌지 표현 (1: 재구매 , 0: 최초 구매) | int64 | X |

## (2) OPT

- orders.csv 파일의 eval_set 칼럼값이 train 인 주문내역에 대한 데이터
- 구매 상품, 장바구니에 담은 순서, 재구매한 상품인지에 대한 정보
- 1384617 rows , 4 columns 으로 구성

| Aa column 명 | 기능 설명 | type | NaN값 여부 |
|---|---|---|---|
| order_id | 주문 ID | int64 | X |
| product_id | 주문한 상품ID | int64 | X |
| add_to_cart_order | 장바구니에 담은 순서 | int64 | X |
| reordered | 재구매한 상품인지 아닌지 표현 (1: 재구매 , 0: 최초 구매) | int64 | X |

# ♉ 가설 검증

## 1 . 제일 많이 판매되는 상품은?

### (1) OPP & OPT concat 하기

- OPP & OPT은 주문 세부 내역을 분할한 것이기 때문에 concat을 실행해 Dataframe을 병합해 order_info DataFrame을 생성
- 33819106 row, 4 columns 으로 구성

**Order_info**

| Aa column 명 | 기능 설명 | type | NaN값 여부 |
|---|---|---|---|
| order_id | 주문 ID | int64 | X |
| product_id | 주문한 상품ID | int64 | X |
| add_to_cart_order | 장바구니에 담은 순서 | int64 | X |
| reordered | 재구매한 상품인지 아닌지 표현 (1: 재구매 , 0: 최초 구매) | int64 | X |

### (2) Order_info DataFrame의 결측치 탐색

- order_info DataFrame은 33819106 개의 row, 4개의 column을 가짐
- 결측치가 없음을 볼 수 있으므로

```
Int64Index: 33819106 entries, 0 to 32434488
Data columns (total 4 columns):
order_id              int64
product_id            int64
add_to_cart_order     int64
reordered             int64
dtypes: int64(4)
```

### (3) groupby () 통해 요약 변수 생성 (Total_orders)

- 각 product 별 총 주문량을 알기 위해 groupby를 수행하고 그 결과를 'Total_orders' 라는 파생변수에 저장
- 결과를 grouped DataFrame에 저장
- 49685의 row, 2개의 column 가짐

| | product_id | Total_orders |
|---|---|---|
| 0 | 1 | 1928 |
| 1 | 2 | 94 |
| 2 | 3 | 283 |
| 3 | 4 | 351 |
| 4 | 5 | 16 |

▶ grouped.head() 결과

### (4) Total_orders가 높은 순서대로 정렬 수행

- 제일 많이 주문된 상품을 알아야 하므로 Total_orders에 대해 sort_values()를 수행한다.

| | product_id | Total_orders |
|---|---|---|
| 24849 | 24852 | 491291 |
| 13173 | 13176 | 394930 |
| 21134 | 21137 | 275577 |
| 21900 | 21903 | 251705 |
| 47205 | 47209 | 220877 |

▶ grouped.head() 결과

### (5) 판매되는 상품의 이름이 무엇인지 알기 위해 Merge 수행

- 상품명 정보가 있는 products Dataframe과 (4)번의 DataFrame을 product_id를 기준으로 Merge
- orderbest.csv 이름으로 DataFrame을 저장
- 49685 rows × 5 columns 가짐

| | product_id | Total_orders | product_name | aisle_id | department_id |
|---|---|---|---|---|---|
| 0 | 24852 | 491291 | Banana | 24 | 4 |
| 1 | 13176 | 394930 | Bag of Organic Bananas | 24 | 4 |
| 2 | 21137 | 275577 | Organic Strawberries | 24 | 4 |
| 3 | 21903 | 251705 | Organic Baby Spinach | 123 | 4 |
| 4 | 47209 | 220877 | Organic Hass Avocado | 24 | 4 |

▶ orderbest.head() 결과

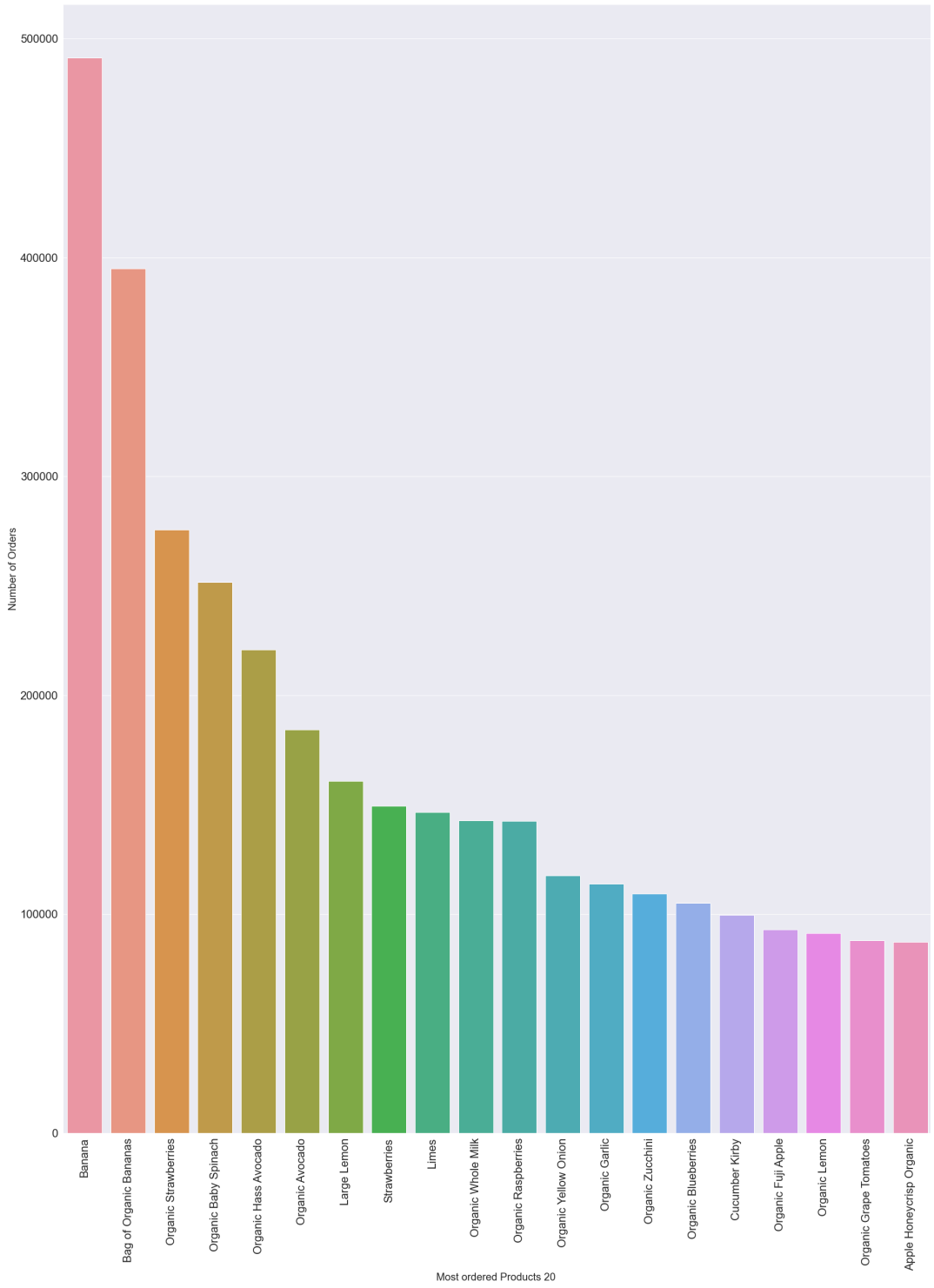
### (6) Orderbest DataFrame의 결측치 확인

- Orderbest DataFrame 의 모든 column에 대해 결측치 여부 확인
- 결측치가 없음을 볼 수 있음

```
product_id      0
Total_orders    0
product_name    0
aisle_id        0
department_id   0
dtype: int64
```

### (7) 주문량이 많은 상위 20개의 product 추출하고 시각화

- 상위 20개의 데이터만 추출해 막대그래프 형식으로 시각화




▶ 주문량 Best 20 상품

### ✔️ 분석 결과

- 주문량 상위 20개의 제품과 그 주문량을 출력해본 결과 바나나와 유기농 바나나가 1,2위로 다른 제품에 비해 압도적으로 주문량이 많음을 볼 수 있다.

## 2. 제품 카테고리 중 주문량이 제일 많은 카테고리는?

### (0) 데이터 준비 / 결측치 확인

- 각 제품에 대한 주문량과 카테고리 정보가 있는 orderbest DataFrame 이용

| | product_id | Total_orders | product_name | aisle_id | department_id |
|---|---|---|---|---|---|
| 0 | 24852 | 491291 | Banana | 24 | 4 |
| 1 | 13176 | 394930 | Bag of Organic Bananas | 24 | 4 |
| 2 | 21137 | 275577 | Organic Strawberries | 24 | 4 |
| 3 | 21903 | 251705 | Organic Baby Spinach | 123 | 4 |
| 4 | 47209 | 220877 | Organic Hass Avocado | 24 | 4 |

▶ orderbest.head()의 결과

- orderbest는 총 49685개의 row, 5개의 column으로 이루어져있음
- 결측값은 존재하지 않고  object type인 product_name을 제외하고 4개의 column의 값은 전부 int값을 가짐

**Order_best**

| Aa column 명 | 기능 설명 | type | NaN값 여부 |
|---|---|---|---|
| product_id | 주문한 상품ID | int64 | X |
| Total_orders | 상품이 주문된 횟수 | int64 | X |
| product_name | 상품 이름 | object | X |
| aisle_id | 상품 세부 카테고리 ID | int64 | X |
| department_id | 상품 카테고리 ID | int64 | |

### (1) Department DataFrame과의 Merge 수행

- 각 제품 카테고리 번호(department_id)의 카테고리 명을 알기 위해 Department DataFrame과 'department_id'를 기준으로 merge수행
- merge한 결과를 product_specific DataFrame에 저장

| | product_id | Total_orders | product_name | aisle_id | department_id | department |
|---|---|---|---|---|---|---|
| 0 | 24852 | 491291 | Banana | 24 | 4 | produce |
| 1 | 13176 | 394930 | Bag of Organic Bananas | 24 | 4 | produce |
| 2 | 21137 | 275577 | Organic Strawberries | 24 | 4 | produce |
| 3 | 21903 | 251705 | Organic Baby Spinach | 123 | 4 | produce |
| 4 | 47209 | 220877 | Organic Hass Avocado | 24 | 4 | produce |

▶ product_specific .head()의 결과

### (2) Merge한 DataFrame (product_specific)의 속성, 결측치 파악

- 총 49685개의 row와 6개의 column으로 이루어져 있음
- 'product_name', 'department' column은 datatype이 object이며 나머지 4개의 column은 모두 int값을 가짐
- 결측치는 존재하지 않음

```
Data columns (total 6 columns):
product_id       49685 non-null int64
Total_orders     49685 non-null int64
product_name     49685 non-null object
aisle_id         49685 non-null int64
department_id    49685 non-null int64
department       49685 non-null object
dtypes: int64(4), object(2)
```

▶ product_specific.info()의 결과

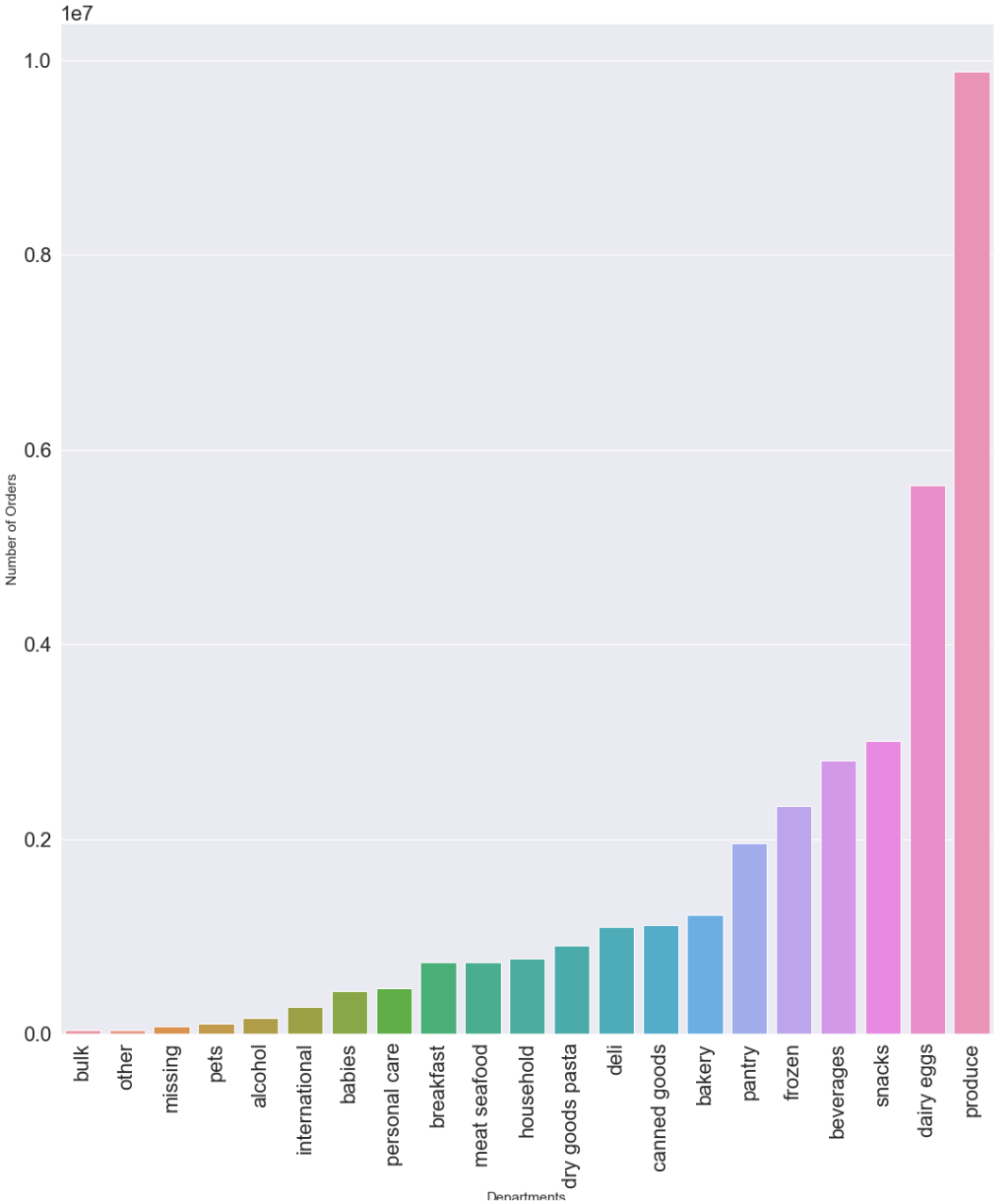
### (3) groupby() 수행하여 요약변수 생성 (total_oreder_num)

- Department 별 주문량의 총 합을 계산해 'total_order_num' 변수를 생성함
  - 상품 카테고리 별 제품의 주문량을 의미
- 그 결과를 'department_product_number' DataFrame에 저장

| | department | total_order_num |
|---|---|---|
| 0 | alcohol | 159294 |
| 1 | babies | 438743 |
| 2 | bakery | 1225181 |
| 3 | beverages | 2804175 |
| 4 | breakfast | 739069 |
| 5 | bulk | 35932 |
| 6 | canned goods | 1114857 |
| 7 | dairy eggs | 5631067 |
| 8 | deli | 1095540 |
| 9 | dry goods pasta | 905340 |
| 10 | frozen | 2336858 |
| 11 | household | 774652 |
| 12 | international | 281155 |
| 13 | meat seafood | 739238 |
| 14 | missing | 77396 |
| 15 | other | 38086 |
| 16 | pantry | 1956819 |
| 17 | personal care | 468693 |
| 18 | pets | 102221 |
| 19 | produce | 9888378 |
| 20 | snacks | 3006412 |

▶ department_product_number

### (4) 카테고리 별 총 주문 갯수를 시각화




## ✔️ 분석 결과

- 히스토그램을 확인해보면 produce 카테고리에 속한 제품들이 제일 많이 주문되었음을 알 수 있음
- 또한, bulk에 해당하는 상품들이 제일 적게 판매되었음을 알 수 있음

## 🚨 Warning!

- produce 카테고리의 주문량이 제일 많다고 해서 그 제품들이 인기가 많다고 보긴 어려움
  - 각 카테고리에 해당하는 제품의 갯수 자이가 있을 수 있기 때문!
- 각 카테고리 별 제품의 갯수 확인해보기

## (5) 각 카테고리 별 상품의 갯수 확인하기

### ✔ 5-1) product_specific Dataframe 확인

| | product_id | Total_orders | product_name | aisle_id | department_id | department |
|---|---|---|---|---|---|---|
| 0 | 24852 | 491291 | Banana | 24 | 4 | produce |
| 1 | 13176 | 394930 | Bag of Organic Bananas | 24 | 4 | produce |
| 2 | 21137 | 275577 | Organic Strawberries | 24 | 4 | produce |
| 3 | 21903 | 251705 | Organic Baby Spinach | 123 | 4 | produce |
| 4 | 47209 | 220877 | Organic Hass Avocado | 24 | 4 | produce |

▶ product_specific .head()의 결과

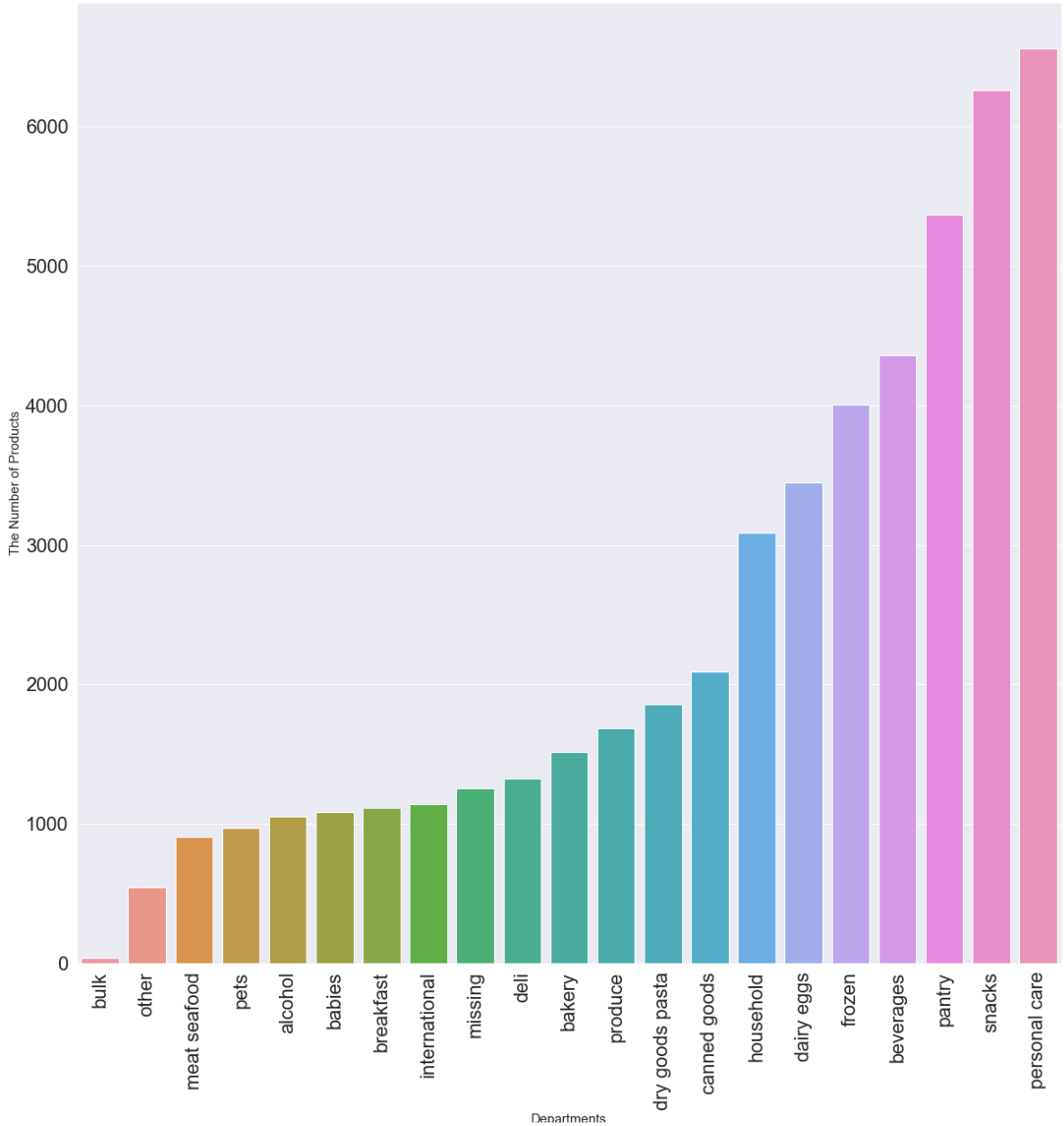
### ✔ 5-2) groupby사용해 요약변수 생성 ( product_num )

- department별 판매되는 상품의 갯수를 구하기 위해 department을 기준으로 product_id를 count해서 'product_num' 요약변수 생성
- product_num을 기준으로 오름차순으로 정렬
- 그 결과를 department_cnt DataFrame 저장

| | department | product_num |
|---|---|---|
| 5 | bulk | 38 |
| 15 | other | 548 |
| 13 | meat seafood | 907 |
| 18 | pets | 972 |
| 0 | alcohol | 1054 |
| 1 | babies | 1081 |
| 4 | breakfast | 1114 |
| 12 | international | 1139 |
| 14 | missing | 1258 |
| 8 | deli | 1322 |
| 2 | bakery | 1516 |
| 19 | produce | 1684 |
| 9 | dry goods pasta | 1858 |
| 6 | canned goods | 2092 |
| 11 | household | 3085 |
| 7 | dairy eggs | 3449 |
| 10 | frozen | 4007 |
| 3 | beverages | 4364 |
| 16 | pantry | 5370 |
| 20 | snacks | 6264 |
| 17 | personal care | 6563 |

▶ department_cnt 의 결과

## ✔ 5-3) 시각화




## ✔ 분석 결과

- personal care 카테고리의 제품이 6563개로 제일 많고,  snacks 카테고리에 포함된 제품의 갯수가 6264개로 두번째로 많음
- 또한, bulk 카테고리가 38개로 제일 적은 상품을 가지고 있음을 볼 수 있음
- 각 카테고리의 상품 갯수 대비 판매량을 비교하여 Instacart Market의 제일 인기있는 상품 카테고리를 알아보자.

### (6) 상품 카테고리에 속하는 상품 갯수 대비 주문량 비교

## ✔ 6-1) 사용 데이터 확인하기

- department_cnt
  - 각 상품 카테고리에 속하는 상품의 갯수 DataFrame
  - 총 21개의 row, 2개의 column을 가짐

| | department | product_num |
|---|---|---|
| 5 | bulk | 38 |
| 15 | other | 548 |
| 13 | meat seafood | 907 |
| 18 | pets | 972 |
| 0 | alcohol | 1054 |

▶ department_cnt.head()의 결과

- department_product_number
  - 각 상품 카테고리 별 총 주문량 DataFrame
  - 총 21개의 row, 2개의 column을 가짐

| | department | total_order_num |
|---|---|---|
| 5 | bulk | 35932 |
| 15 | other | 38086 |
| 14 | missing | 77396 |
| 18 | pets | 102221 |
| 0 | alcohol | 159294 |

▶ department_product_number.head()의 결과

## ✔ 6-2) 2개의 DataFrame을 Merge

- 상품 갯수 대비 주문량을 비교하기 위해 department_cnt & department_product_number를 'department'를 기준으로 Merge 수행
- 그 결과를 department_product DataFrame에 저장

| | department | product_num | total_order_num |
|---|---|---|---|
| 0 | bulk | 38 | 35932 |
| 1 | other | 548 | 38086 |
| 2 | meat seafood | 907 | 739238 |
| 3 | pets | 972 | 102221 |
| 4 | alcohol | 1054 | 159294 |

▶ department_product.head()의 결과

## ✔ 6-3) Department_product DataFrame의 결측치 확인

```
Int64Index: 21 entries, 0 to 20
Data columns (total 3 columns):
department         21 non-null object
product_num        21 non-null int64
total_order_num    21 non-null int64
dtypes: int64(2), object(1)
```

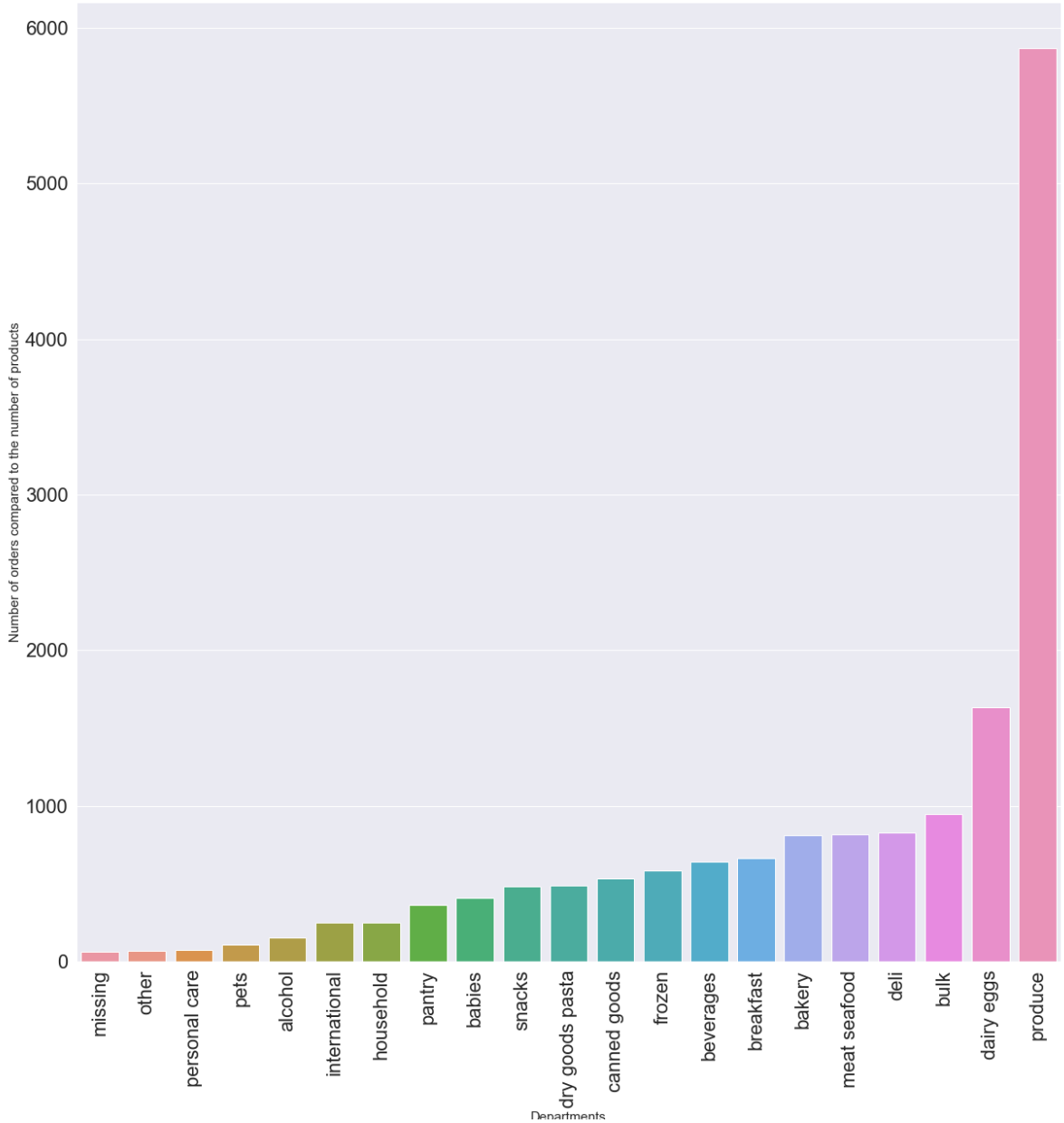
## ✔ 6-4) 상품 갯수 대비 주문량에 대한 파생변수 생성 & 오름차순 정렬

- 상품 갯수 대비 주문량을 구하기 위해 total_order_num/product_num 수행해 product_per_order 파생변수 생성
- product_per_order에 대해 오름차순으로 정렬수행

| | department | product_num | total_order_num | product_per_order |
|---|---|---|---|---|
| 8 | missing | 1258 | 77396 | 61.523052 |
| 1 | other | 548 | 38086 | 69.500000 |
| 20 | personal care | 6563 | 468693 | 71.414445 |
| 3 | pets | 972 | 102221 | 105.165638 |
| 4 | alcohol | 1054 | 159294 | 151.132827 |

▶ product_per_order.head()의 결과

## ✔ 6-5) 시각화




## ✔ 분석 결과

- 각 상품의 갯수 대비 주문량에 대한 히스토그램 결과를 보면 produce 카테고리의 상품이 상품의 갯수에 비해 주문되는 양이 제일 많다는 것을 알 수 있음
- 위의 히스토그램들과 비교해봤을 때, bulk 카테고리에 속하는 상품의 갯수가 제일 적지만 상품 갯수 대비 주문량은 3위임을 알 수 있음.
- 이는 다른 카테고리의 상품에 비해 produce 카테고리의 상품이 고객들에게 인기가 많다는 것을 의미함. 그러므로  produce 카테고리의 상품들에 대해 할인을 하는 등의 마케팅을 진행 할 수 있음.