In [5]:
```
!pip install colorama
```

```
Requirement already satisfied: colorama in /usr/local/lib/python3.10/di
st-packages (0.4.6)
```

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import gc

from tqdm.auto import tqdm
import math
from sklearn.model_selection import KFold, StratifiedKFold, train_test_s
import warnings
warnings.filterwarnings('ignore')




tqdm.pandas()

rc = {
    "axes.facecolor": "#FFF9ED",
    "figure.facecolor": "#FFF9ED",
    "axes.edgecolor": "#000000",
    "grid.color": "#EBEBE7",
    "font.family": "serif",
    "axes.labelcolor": "#000000",
    "xtick.color": "#000000",
    "ytick.color": "#000000",
    "grid.alpha": 0.4
}

sns.set(rc=rc)

from colorama import Style, Fore
red = Style.BRIGHT + Fore.RED
blu = Style.BRIGHT + Fore.BLUE
mgt = Style.BRIGHT + Fore.MAGENTA
gld = Style.BRIGHT + Fore.YELLOW
res = Style.RESET_ALL
```

In [2]:
```
data = pd.read_csv('amazon_products.csv')
```

In [3]:
```
# summary table function
def summary(df):
    print(f'data shape: {df.shape}')
    summ = pd.DataFrame(df.dtypes, columns=['data type'])
    summ['#missing'] = df.isnull().sum().values
    summ['%missing'] = df.isnull().sum().values / len(df)* 100
    summ['#unique'] = df.nunique().values
    desc = pd.DataFrame(df.describe(include='all').transpose())
    summ['min'] = desc['min'].values
    summ['max'] = desc['max'].values
    summ['first value'] = df.loc[0].values
    summ['second value'] = df.loc[1].values
    summ['third value'] = df.loc[2].values

    return summ
```

In [4]:
```
summary(data)
```

```
data shape: (1048575, 12)
```

Out[4]:

	data type	#missing	%missing	#unique	min	max	
asin	object	0	0.0	1048575	NaN	NaN	
title	object	0	0.0	1023107	NaN	NaN	Sion Softside Expandable
imgUrl	object	0	0.0	1011144	NaN	NaN	h amazon.com/images/
productURL	object	0	0.0	1048575	NaN	NaN	https://www.amazon.com/d
stars	float64	0	0.0	41	0	5	
reviews	int64	0	0.0	10074	0	292474	
price	float64	0	0.0	26559	0	19731.8	
listPrice	float64	0	0.0	12292	0	999.99	
category_id	int64	0	0.0	190	1	270	
Category_name	object	0	0.0	190	NaN	NaN	
isBestSeller	bool	0	0.0	2	NaN	NaN	
boughtInLastMonth	int64	0	0.0	30	0	100000	

In [ ]:
```
# # select numerical and categorical variables respectively.
# num_cols = data.select_dtypes(include=['float64','int64']).columns.tol

# cat_cols = data.select_dtypes(include=['object']).columns.tolist()
# cat_cols.remove('asin')
# cat_cols.remove('imgUrl')
# cat_cols.remove('productURL')
# all_features = num_cols + cat_cols
```

In [5]:
```
products_data = data.drop( columns = ['asin', 'imgUrl', 'productURL','ca
products_data.head()
```

Out[5]:

	title	stars	reviews	price	listPrice	Category_name	isBestSeller	boughtInLastMonth
0	Sion Softside Expandable Roller Luggage, Black...	4.5	0	139.99	0.00	Suitcases	False	2000
1	Luggage Sets Expandable PC+ABS Durable Suitcas...	4.5	0	169.99	209.99	Suitcases	False	1000
2	Platinum Elite Softside Expandable Checked Lug...	4.6	0	365.49	429.99	Suitcases	False	300
3	Freeform Hardside Expandable with Double Spinn...	4.6	0	291.59	354.37	Suitcases	False	400
4	Winfield 2 Hardside Expandable Luggage with Sp...	4.5	0	174.99	309.99	Suitcases	False	400

In [6]:
```
products_data.isnull().sum()
products_data.shape
```

Out[6]:
```
(1048575, 8)
```

In [7]:
```
products_data.describe(include = 'all')
```

Out[7]:

	title	stars	reviews	price	listPrice	Category_name	isBe
count	1048575	1.048575e+06	1.048575e+06	1.048575e+06	1.048575e+06	1048575	1
unique	1023107	NaN	NaN	NaN	NaN	190	
top	Men's Ultraboost 23 Running Shoe	NaN	NaN	NaN	NaN	Girls' Clothing	
freq	83	NaN	NaN	NaN	NaN	28619	1
mean	NaN	3.993607e+00	1.774382e+02	4.476936e+01	1.264740e+01	NaN	
std	NaN	1.350485e+00	1.624363e+03	1.405284e+02	4.768020e+01	NaN	
min	NaN	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	NaN	
25%	NaN	4.100000e+00	0.000000e+00	1.199000e+01	0.000000e+00	NaN	
50%	NaN	4.400000e+00	0.000000e+00	1.999000e+01	0.000000e+00	NaN	
75%	NaN	4.600000e+00	0.000000e+00	3.599000e+01	0.000000e+00	NaN	
max	NaN	5.000000e+00	2.924740e+05	1.973181e+04	9.999900e+02	NaN	

```
# Men's Sneaker is the most frequent product title in the product card
(89 times)

The median rating is 4.4

The median number of reviews is 0, and the mean is 181. 75% of products
have no reviews. It would be interesting to see how the reviews are
distributed and how many products have reviews.

The average product price is 43, the median is 20, and the
interquartile range is 12-36. The mean does not fall within the
interquartile range, which is due to the large spread of prices. The
minimum price is 0, which is not possible, so these values need to be
checked and possibly removed.

75% of products have no discount, the average discount is 1.2, and the
maximum is 999, which is a very large spread.

The vast majority of products do not have bestseller status.

50% of products have 0 sales in the past month.
```

In [8]:
```
# lines with zero price

products_data[products_data.price == 0]
```

Out[8]:

	title	stars	reviews	price	listPrice	Category_name	isBestSeller	boughtInLastM
177	Airconic Hardside Expandable Luggage with Spin...	3.8	0	0.0	0.0	Suitcases	False	
180	4 KIX 2.0 Softside Expandable Luggage with Spi...	4.5	0	0.0	0.0	Suitcases	False	
252	OCEAN HOLIDAY Luggage with Double Spinner Whee...	0.0	0	0.0	0.0	Suitcases	False	
328	Merge Short Trip Expandable Packing Case Mediu...	3.9	0	0.0	0.0	Suitcases	False	
445	Suitcase Caster, 1 Pair Luggage Swivel Wheels ...	4.0	0	0.0	0.0	Suitcases	False	
...	...	...	...	...	...	...	...	...
1048353	Waverly Inspirations Fat Quarter Stack 5 Cotto...	4.5	0	0.0	0.0	Craft & Hobby Fabric	False	
1048362	Nicole's Prints Las Elegantes Bright Smoke, Fa...	4.8	0	0.0	0.0	Craft & Hobby Fabric	False	
1048465	Spoonflower Fabric - Mini Slaying Nezuko ASA P...	4.5	0	0.0	0.0	Craft & Hobby Fabric	False	
1048488	Cotton Cancer Awareness, Black/Pink Cut by The...	4.6	0	0.0	0.0	Craft & Hobby Fabric	False	
1048522	Flannel Giraffes Giraffe Animals Kids Children...	4.5	0	0.0	0.0	Craft & Hobby Fabric	False	

24597 rows × 8 columns

In [9]:
```
products_data.groupby('Category_name').count()
```

Out[9]:

Category_name	title	stars	reviews	price	listPrice	isBestSeller	boughtInLastMonth
Accessories & Supplies	2426	2426	2426	2426	2426	2426	2426
Additive Manufacturing Products	7619	7619	7619	7619	7619	7619	7619
Arts, Crafts & Sewing Storage	7592	7592	7592	7592	7592	7592	7592
Automotive Enthusiast Merchandise	253	253	253	253	253	253	253
Automotive Exterior Accessories	8536	8536	8536	8536	8536	8536	8536
...	...	...	...	...	...	...	...
Women's Jewelry	17005	17005	17005	17005	17005	17005	17005
Xbox 360 Games, Consoles & Accessories	3809	3809	3809	3809	3809	3809	3809
Xbox One Games, Consoles & Accessories	3582	3582	3582	3582	3582	3582	3582
Xbox Series X & S Consoles, Games & Accessories	5645	5645	5645	5645	5645	5645	5645
eBook Readers & Accessories	582	582	582	582	582	582	582

190 rows × 7 columns

```
Lines with a zero price are present in all 190 product categories, so
they can be deleted, this will not spoil the representativeness of the
sample.
```

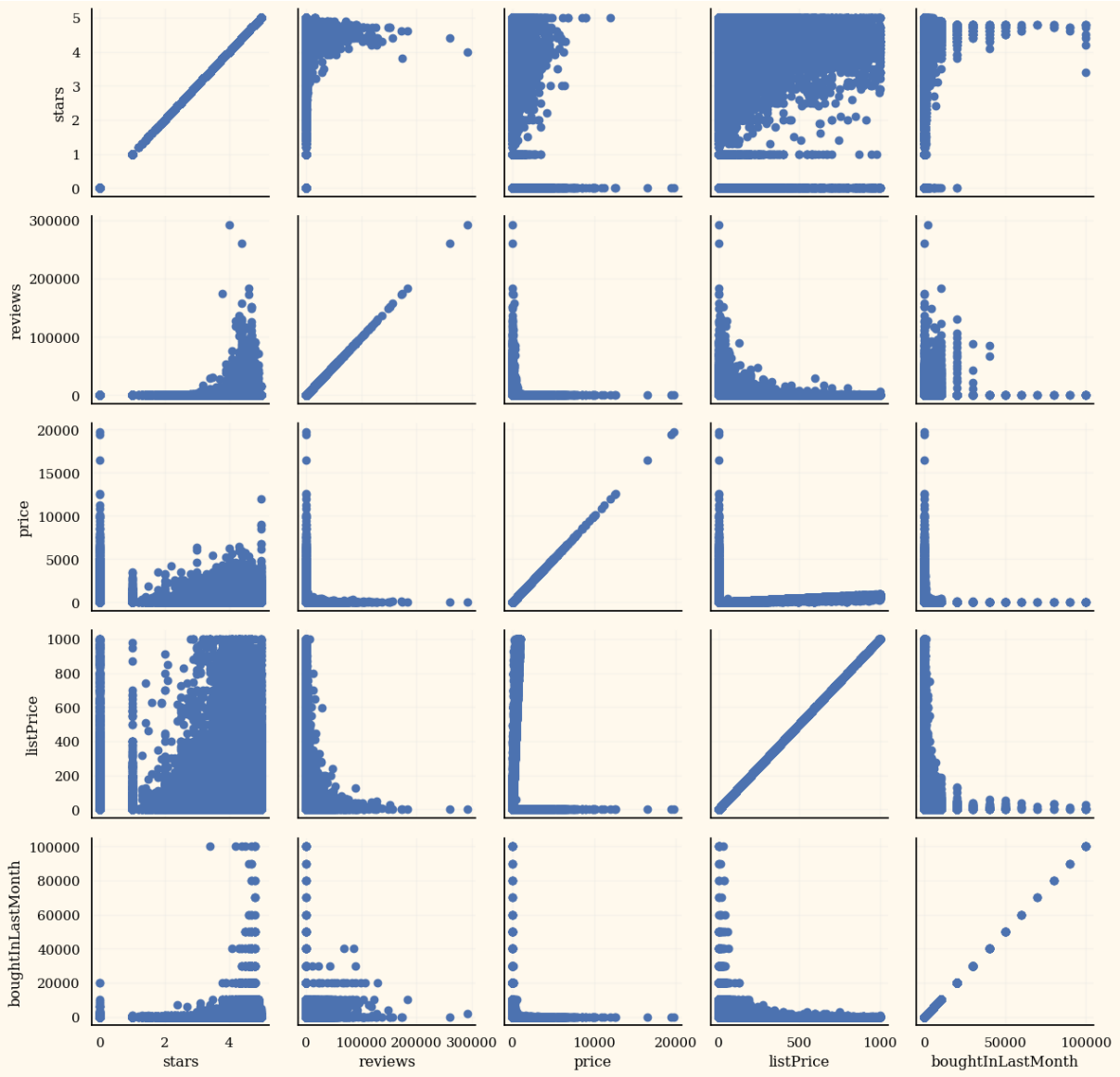
In [10]:
```
products_data = products_data[products_data.price > 0]
products_data.shape
```

Out[10]: (1023978, 8)

In [19]:
```
#Let's look at how quantitative variables are distributed.

products_subset = products_data.copy()
products_subset.drop(['title', 'Category_name', 'isBestSeller'], axis=1,

pair_grid_plot_1 = sns.PairGrid(products_subset)
pair_grid_plot_1.map(plt.scatter);
```



The higher the rating, the more reviews the product has. The higher the rating of a product, the more it was bought last month. The more reviews, the fewer goods were bought last month, which is strange, we need to look at reviews separately. Price and rating on the graph do not have a strong relationship. There is no rating at all for goods with a high price, probably because goods with a high cost are bought less often.

What is the best selling product?

In [11]:
```
products_data[products_data.boughtInLastMonth == products_data.boughtInLa
```

Out[11]:

	title	stars	reviews	price	listPrice	Category_name	isBestSeller	boughtInLastMo
331036	Bounty Quick Size Paper Towels, White, 8 Famil...	4.8	0	24.42	0.00	Household Cleaning Supplies	True	100
331037	Amazon Brand - Presto! Flex-a-Size Paper Towel...	4.7	0	28.28	0.00	Household Cleaning Supplies	False	100
331038	Stardrops - The Pink Stuff - The Miracle All P...	4.4	0	4.99	5.97	Household Cleaning Supplies	True	100
331040	Amazon Basics 2-Ply Paper Towels, Flex-Sheets,...	4.2	0	22.86	0.00	Household Cleaning Supplies	False	100
466031	Hismile v34 Colour Corrector, Tooth Stain Remo...	3.4	0	20.69	29.00	Oral Care Products	True	100
1009205	LMNT Zero-Sugar Electrolytes - Variety Salt - ...	4.5	0	25.00	0.00	Sports Nutrition Products	False	100
1016564	Nespresso Capsules VertuoLine, Medium and Dark...	4.8	0	37.50	0.00	Kitchen & Dining	False	100
1016566	Dixie Paper Plates, 8 1/2 inch, Dinner Size Pr...	4.8	0	5.99	6.89	Kitchen & Dining	True	100

```
The most popular products were household items: paper towels, cleaning
supplies, teeth whitening, water electrolytes, mascara, coffee pods,
paper disposable plates, facial essence, acne patches, makeup removal
wipes. That is, goods for the home and personal care.
```

There are no reviews (except for mascara) on these most popular products in the dataset. If you go to the links and look at the product card on the Amazon website, they all have many thousands of reviews. Therefore, errors were made in data collection.

Let's see how many products do not have review data.

In [12]:
```
print(round(len(products_data[products_data.reviews == 0])/len(products_
      '% products do not have review.')
```

```
76.9 % products do not have review.
```

In [13]:
```
# We will drop review column as it is misleading

products_data.drop(['reviews'], axis=1, inplace=True)
products_data.head()
```

Out[13]:

	title	stars	price	listPrice	Category_name	isBestSeller	boughtInLastMonth
0	Sion Softside Expandable Roller Luggage, Black...	4.5	139.99	0.00	Suitcases	False	2000
1	Luggage Sets Expandable PC+ABS Durable Suitcas...	4.5	169.99	209.99	Suitcases	False	1000
2	Platinum Elite Softside Expandable Checked Lug...	4.6	365.49	429.99	Suitcases	False	300
3	Freeform Hardside Expandable with Double Spinn...	4.6	291.59	354.37	Suitcases	False	400
4	Winfield 2 Hardside Expandable Luggage with Sp...	4.5	174.99	309.99	Suitcases	False	400

In [14]:
```
# Let's add an income column
products_data['income'] = products_data.price * products_data.boughtInLa
products_data.head()
```

Out[14]:

	title	stars	price	listPrice	Category_name	isBestSeller	boughtInLastMonth	income
0	Sion Softside Expandable Roller Luggage, Black...	4.5	139.99	0.00	Suitcases	False	2000	279980.0
1	Luggage Sets Expandable PC+ABS Durable Suitcas...	4.5	169.99	209.99	Suitcases	False	1000	169990.0
2	Platinum Elite Softside Expandable Checked Lug...	4.6	365.49	429.99	Suitcases	False	300	109647.0
3	Freeform Hardside Expandable with Double Spinn...	4.6	291.59	354.37	Suitcases	False	400	116636.0
4	Winfield 2 Hardside Expandable Luggage with Sp...	4.5	174.99	309.99	Suitcases	False	400	69996.0

In [15]:
```
# Let's check the most profitable item regardless of category
products_data[products_data.income == products_data.income.max()]
```

Out[15]:

	title	stars	price	listPrice	Category_name	isBestSeller	boughtInLastMonth	in
137848	PlayStation 5 Console (PS5)	4.8	499.99	0.0	Video Games	False	10000	4999

The most profitable product in September is the PlayStation 5 Console (PS5). Let's take a closer look at which products in the video game category turned out to be the most profitable.

In [16]:
```
game_max = products_data[products_data['Category_name'] ==
                         'Video Games'].sort_values('income', ascending =
game_max
```

Out[16]:

	title	stars	price	listPrice	Category_name	isBestSeller	boughtInLastMonth	in
137848	PlayStation 5 Console (PS5)	4.8	499.99	0.00	Video Games	False	10000	4999
137850	Xbox Series X	4.8	489.99	0.00	Video Games	True	10000	4899
137851	Meta Quest 2 — Advanced All-In-One Virtual Rea...	4.7	299.00	0.00	Video Games	True	10000	2990
137852	Nintendo Switch™ with Neon Blue and Neon Red J...	4.8	299.00	0.00	Video Games	True	9000	2691
137857	Xbox Series S – 512GB SSD All-Digital Gaming C...	4.8	298.00	0.00	Video Games	False	5000	1490
137860	Nintendo Switch – OLED Model w/ White Joy-Con	4.8	339.00	0.00	Video Games	False	4000	1356
137858	Xbox Elite Series 2 Core Wireless Controller –...	4.5	99.00	129.99	Video Games	False	9000	891
137867	Meta Quest 2 - Advanced All-In-One Virtual Rea...	4.6	269.99	299.00	Video Games	False	3000	809
137849	Final Fantasy VII Rebirth - Deluxe Edition (PS5)	0.0	99.99	0.00	Video Games	False	8000	799
137862	Xbox Wireless Headset – Xbox Series X|S, Xbox ...	4.5	91.79	99.99	Video Games	False	8000	734

In [17]:
```
video_games = products_data[products_data['Category_name'] == 'Video Game
video_games.describe()
```

Out[17]:

	stars	price	listPrice	boughtInLastMonth	income
count	5149.000000	5149.000000	5149.000000	5149.000000	5.149000e+03
mean	4.309186	55.310511	24.537199	168.712371	1.181722e+04
std	0.849573	117.413128	55.344590	636.029649	1.223241e+05
min	0.000000	2.500000	0.000000	0.000000	0.000000e+00
25%	4.300000	19.990000	0.000000	0.000000	0.000000e+00
50%	4.500000	29.990000	0.000000	0.000000	0.000000e+00
75%	4.700000	51.930000	32.990000	100.000000	3.998000e+03
max	5.000000	4499.990000	999.990000	10000.000000	4.999900e+06

Top 20 Most profitable product categories

In [18]:
```
agg_func = {
    'stars': 'median',
    'price': 'median',
    'listPrice': 'median',
    'isBestSeller': 'sum',
    'boughtInLastMonth': 'sum',
    'income': 'sum'
    }
top_categories = products_data.groupby('Category_name').agg(agg_func).ro


top_categories.style.background_gradient(axis = 0, cmap = 'PuBu')
```

Out[18]:

Category_name	stars	price	listPrice	isBestSeller	boughtInLastMonth	income
Kitchen & Dining	4.600000	16.230000	0.000000	279.000000	10391100	267189588.000000
Hair Care Products	4.500000	14.950000	0.000000	44.000000	7925600	152940697.500000
Home Storage & Organization	4.500000	23.990000	0.000000	118.000000	5338750	138604708.500000
Toys & Games	4.500000	19.990000	0.000000	240.000000	5746350	135394508.500000
Industrial & Scientific	4.600000	14.990000	0.000000	400.000000	7057850	130196201.500000
Household Cleaning Supplies	4.500000	15.570000	0.000000	52.000000	6783500	120567961.500000
Skin Care Products	4.500000	16.950000	0.000000	25.000000	6402700	119996888.500000
Dog Supplies	4.400000	17.960000	0.000000	105.000000	4497100	101942514.500000
Office Electronics	4.400000	46.980000	0.000000	35.000000	1714800	95038114.000000
Health & Household	4.600000	12.020000	7.990000	53.000000	5926000	91824980.000000
Sports & Fitness	4.500000	18.690000	0.000000	480.000000	3983750	91626516.000000
Sports Nutrition Products	4.400000	29.990000	0.000000	36.000000	3023800	90938064.000000
Bedding	4.500000	24.990000	0.000000	54.000000	2706850	86802691.000000
Vacuum Cleaners & Floor Care	4.500000	18.990000	0.000000	14.000000	821950	82695750.000000
Sports & Outdoors	4.600000	17.990000	0.000000	257.000000	2947150	76251321.500000
Heating, Cooling & Air Quality	4.500000	31.990000	0.000000	46.000000	1437300	74840634.000000
Automotive Tools & Equipment	4.500000	24.650000	0.000000	149.000000	1663650	74697831.500000
Health Care Products	4.500000	14.990000	0.000000	50.000000	3701750	68067197.500000
Women's Clothing	4.300000	29.990000	0.000000	163.000000	2787450	67695103.000000
Household Supplies	4.600000	13.990000	0.000000	51.000000	4250500	66198883.500000

The table provides summary characteristics for the 20 most popular categories.

Kitchen & Dining is the leader among all categories: the highest revenue and the largest number of sold products, while the average price is relatively small, 16 dollars.

The next category by revenue is hair care products, with an average price of also 15 dollars.

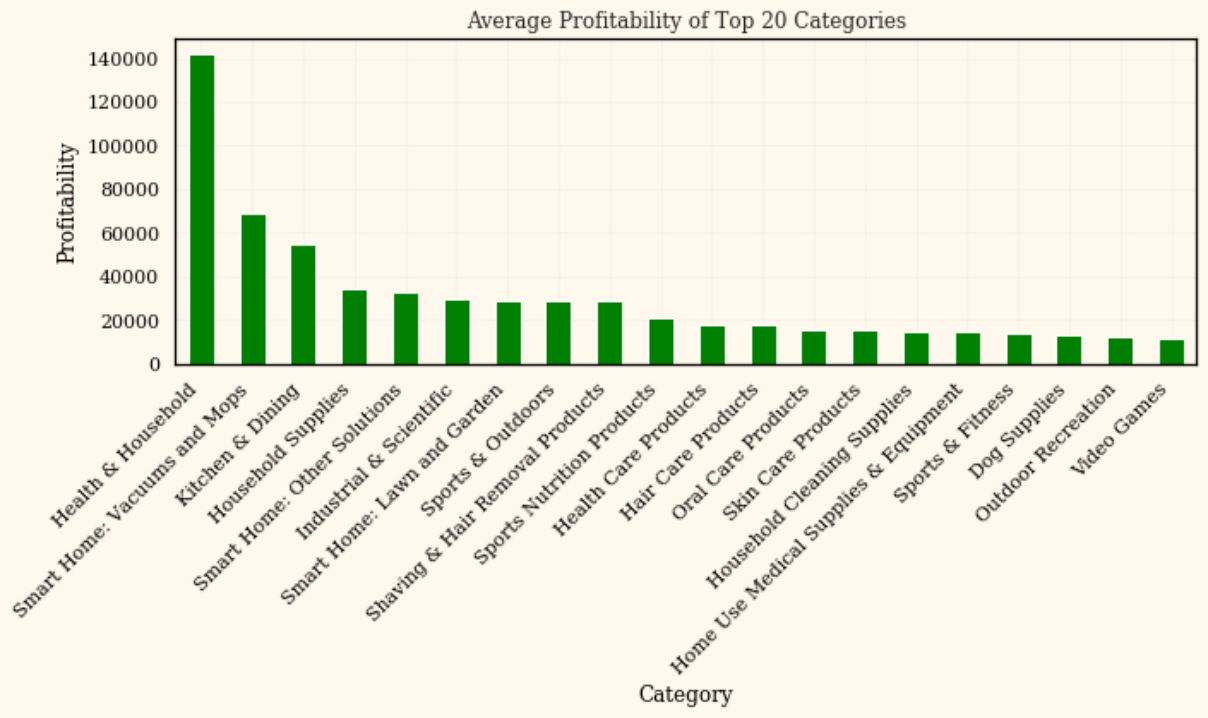
Let's analyze in more detail the most popular products in the kitchen category.

In [19]:
```
products_data['Profitability'] = products_data['income']
top_categories_profitability = products_data.groupby('Category_name')['P
```

In [20]:
```
top_categories_profitability
```

Out[20]:
```
Category_name
Health & Household                          142143.931889
Smart Home: Vacuums and Mops                 69235.890000
Kitchen & Dining                             54932.069901
Household Supplies                           34695.431604
Smart Home: Other Solutions                  33010.224000
Industrial & Scientific                      29509.565163
Smart Home: Lawn and Garden                  29241.204545
Sports & Outdoors                            28904.974033
Shaving & Hair Removal Products              28733.386937
Sports Nutrition Products                    21177.937587
Health Care Products                         18078.936919
Hair Care Products                           17891.986137
Oral Care Products                           15867.825891
Skin Care Products                           15382.244392
Household Cleaning Supplies                  14561.348007
Home Use Medical Supplies & Equipment        14424.648698
Sports & Fitness                             14254.280647
Dog Supplies                                 13187.906145
Outdoor Recreation                           12387.781144
Video Games                                  11817.220528
Name: Profitability, dtype: float64
```

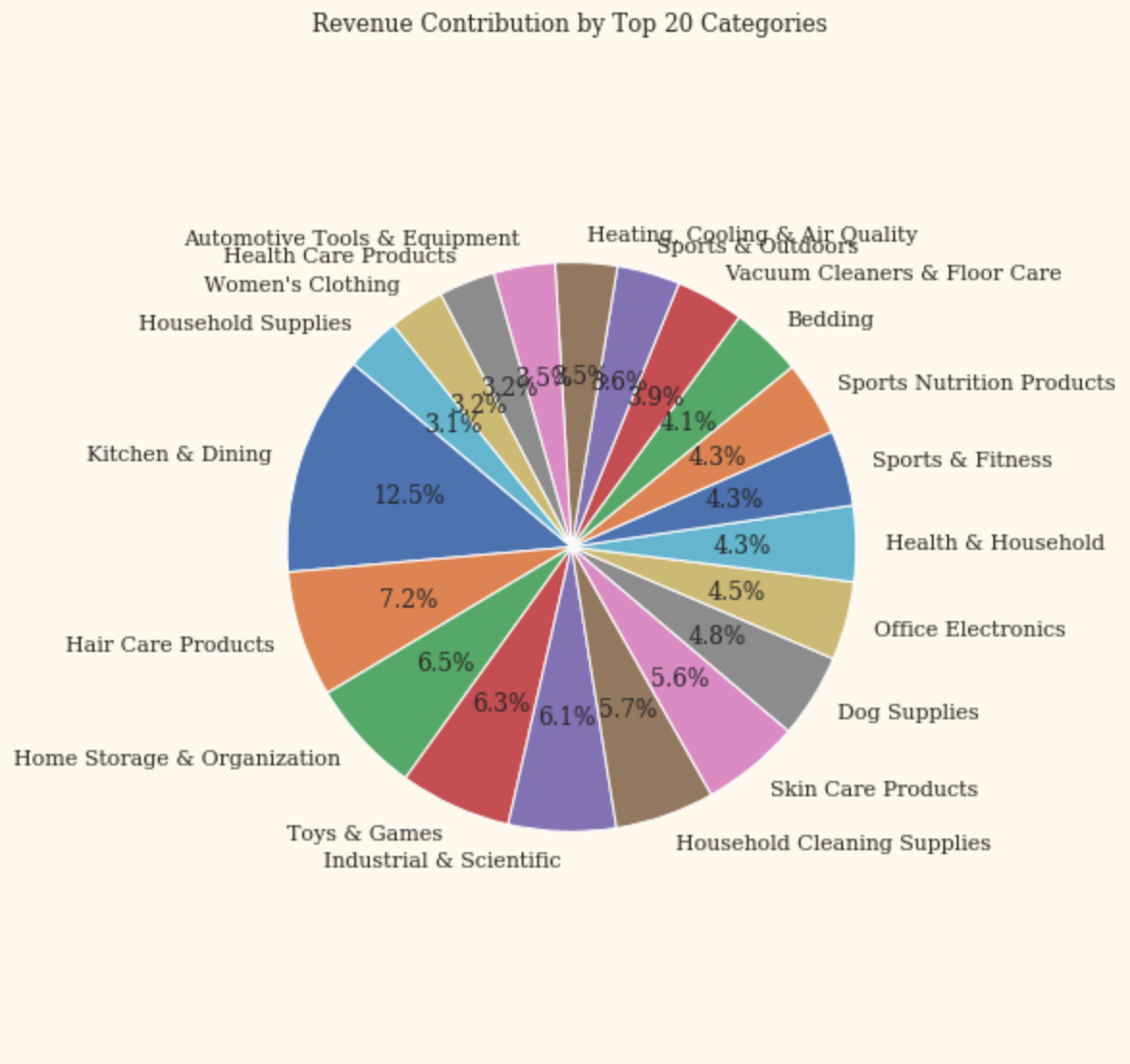
In [21]:
```
plt.figure(figsize=(10, 6))
top_categories_profitability.plot(kind='bar', color='green')
plt.title('Average Profitability of Top 20 Categories')
plt.xlabel('Category')
plt.ylabel('Profitability')
plt.xticks(rotation=45, ha='right')  # Rotate x-labels for better readab
plt.tight_layout()
plt.show()
```



In [22]:
```
total_top_categories_income = top_categories['income'].sum()
top_categories['Revenue_Percentage'] = (top_categories['income'] / total_
print(top_categories[['income', 'Revenue_Percentage']])
```

Category_name	income	Revenue_Percentage
Kitchen & Dining	267189588.0	12.523474
Hair Care Products	152940697.5	7.168501
Home Storage & Organization	138604708.5	6.496557
Toys & Games	135394508.5	6.346092
Industrial & Scientific	130196201.5	6.102441
Household Cleaning Supplies	120567961.5	5.651155
Skin Care Products	119996888.5	5.624388
Dog Supplies	101942514.5	4.778159
Office Electronics	95038114.0	4.454542
Health & Household	91824980.0	4.303939
Sports & Fitness	91626516.0	4.294637
Sports Nutrition Products	90938064.0	4.262368
Bedding	86802691.0	4.068539
Vacuum Cleaners & Floor Care	82695750.0	3.876042
Sports & Outdoors	76251321.5	3.573984
Heating, Cooling & Air Quality	74840634.0	3.507864
Automotive Tools & Equipment	74697831.5	3.501171
Health Care Products	68067197.5	3.190385
Women's Clothing	67695103.0	3.172945
Household Supplies	66198883.5	3.102815

In [23]:
```
plt.figure(figsize=(8, 8))
plt.pie(top_categories['income'], labels=top_categories.index, autopct='%
plt.title('Revenue Contribution by Top 20 Categories')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a c
plt.tight_layout()
plt.show()
```



In [24]:
```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\DIA\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[24]: True

In [25]:
```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\DIA\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[25]:
```
True
```

Most popular words in Kitchen and Dining Category

In [26]:
```
kitchen = products_data[products_data['Category_name'] == 'Kitchen & Din
kitchen['boughtInLastMonth'].describe()
```

Out[26]:
```
count      4864.000000
mean       2136.328125
std        4138.002555
min           0.000000
25%         900.000000
50%        1000.000000
75%        2000.000000
max      100000.000000
Name: boughtInLastMonth, dtype: float64
```

Less than 1000 units were sold for 50% of the products. To analyze the popularity of words in product names, we will only consider items that were purchased more than 999 times.

In [29]:
```
# Importing the built-in module for string manipulation
import string

# Importing regular expressions
import re

# Importing the nltk library for text processing
import nltk

# Importing statistics module
from nltk.probability import FreqDist

# From the text processing library, importing the module for word tokeni
from nltk import word_tokenize

# Importing the module with stop words
from nltk.corpus import stopwords

# Importing the library for creating word clouds
from wordcloud import WordCloud
```

In [30]:
```
# Text preparation function for tokenization

def prep(text):

    # Convert the list to a string with spaces between words
    text = ' '.join(str(ch) for ch in text)

    # Convert all characters to lowercase
    text = text.lower()

    # Add quotes and ellipsis to the standard punctuation marks
    spec_chars = string.punctuation + '«»\t—…’'

    # Remove punctuation marks from the text
    text = "".join([ch for ch in text if ch not in spec_chars])

    # Remove digits from the text
    text = "".join([ch for ch in text if ch not in string.digits])

    return text
```

In [37]:
```
from collections import Counter
import string
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist


# Filter 'kitchen' DataFrame for 'boughtInLastMonth' > 499
high_bought = kitchen[kitchen['boughtInLastMonth'] > 999]

# Copying titles of products into a list
text = high_bought['title'].tolist()

# Applying the text preparation function
text = prep(text)

# Adding stop words
english_stopwords = stopwords.words("english")
words = ['pack', 'oz', 'inch', 'pcs', 'count', 'x', 'kitchen']
english_stopwords.extend(words)

# Tokenizing the text
text_tokens = word_tokenize(text)

# Rearranging tokens, excluding stop words
text_tokens = [token.strip() for token in text_tokens if token not in eng

# Converting tokens to a textual format
text = nltk.Text(text_tokens)

# Calculating word frequencies
fdist_sw = FreqDist(text)

# Displaying the most common words
print(fdist_sw.most_common(20))
```

```
[('coffee', 944), ('steel', 662), ('stainless', 606), ('tea', 571), ('s
et', 504), ('black', 450), ('food', 374), ('roast', 364), ('cooking', 3
15), ('baking', 312), ('cup', 311), ('maker', 286), ('silicone', 283),
('ice', 277), ('pods', 257), ('nonstick', 247), ('bags', 219), ('safe',
216), ('brush', 215), ('free', 212)]
```

Most popular words in Hair Care Products

In [36]:
```
HairCare = products_data[products_data['Category_name'] == 'Hair Care Pr
HairCare['boughtInLastMonth'].describe()
```

Out[36]:
```
count     8548.000000
mean       927.187646
std       1295.464480
min          0.000000
25%        300.000000
50%        500.000000
75%       1000.000000
max      20000.000000
Name: boughtInLastMonth, dtype: float64
```

Less than 500 units were sold for 50% of the products. To analyze the popularity of words in product names, we will only consider items that were purchased more than 499 times.

In [38]:
```
from collections import Counter
import string
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist


# Filter 'HairCare' DataFrame for 'boughtInLastMonth' > 499
high_bought = HairCare[HairCare['boughtInLastMonth'] > 499]

# Copying titles of products into a list
text = high_bought['title'].tolist()

# Applying the text preparation function
text = prep(text)

# Adding stop words
english_stopwords = stopwords.words("english")
words = ['pack', 'oz', 'inch', 'pcs', 'count', 'x', 'HairCare']
english_stopwords.extend(words)

# Tokenizing the text
text_tokens = word_tokenize(text)

# Rearranging tokens, excluding stop words
text_tokens = [token.strip() for token in text_tokens if token not in eng

# Converting tokens to a textual format
text = nltk.Text(text_tokens)

# Calculating word frequencies
fdist_sw = FreqDist(text)

# Displaying the most common words
print(fdist_sw.most_common(20))
```

```
[('hair', 7449), ('shampoo', 1219), ('women', 1136), ('conditioner', 10
00), ('oil', 845), ('fl', 785), ('clips', 730), ('black', 543), ('fre
e', 529), ('brush', 528), ('men', 522), ('styling', 511), ('color', 50
6), ('dry', 479), ('girls', 449), ('curly', 428), ('set', 412), ('scal
p', 412), ('natural', 405), ('accessories', 367)]
```

Most popular words in Home Storage & Organization

In [39]:
```
HomeStorage = products_data[products_data['Category_name'] == 'Home Stora
HomeStorage['boughtInLastMonth'].describe()
```

Out[39]:
```
count    15099.000000
mean       353.583019
std        712.508464
min          0.000000
25%          0.000000
50%        100.000000
75%        400.000000
max      10000.000000
Name: boughtInLastMonth, dtype: float64
```

Less than 100 units were sold for 50% of the products. To analyze the popularity of words in product names, we will only consider items that were purchased more than 99 times.

In [40]:
```
from collections import Counter
import string
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist


# Filter 'HomeStorage' DataFrame for 'boughtInLastMonth' > 99
high_bought = HomeStorage[HomeStorage['boughtInLastMonth'] > 99]

# Copying titles of products into a list
text = high_bought['title'].tolist()

# Applying the text preparation function
text = prep(text)

# Adding stop words
english_stopwords = stopwords.words("english")
words = ['pack', 'oz', 'inch', 'pcs', 'count', 'x', 'HomeStorage']
english_stopwords.extend(words)

# Tokenizing the text
text_tokens = word_tokenize(text)

# Rearranging tokens, excluding stop words
text_tokens = [token.strip() for token in text_tokens if token not in eng

# Converting tokens to a textual format
text = nltk.Text(text_tokens)

# Calculating word frequencies
fdist_sw = FreqDist(text)

# Displaying the most common words
print(fdist_sw.most_common(20))
```

```
[('storage', 4817), ('organizer', 3326), ('kitchen', 2005), ('rack', 17
86), ('holder', 1598), ('bag', 1481), ('food', 1338), ('containers', 13
04), ('box', 1302), ('black', 1289), ('plastic', 1225), ('closet', 121
6), ('clothes', 1191), ('bags', 1178), ('lunch', 1118), ('large', 969),
('steel', 898), ('clear', 887), ('shoe', 870), ('tool', 853)]
```

In [ ]:

In [ ]: