MAGNUM

Next Career Challenge

Primary Mission

Are you a Sunrise or Sunset ice cream lover?

To answer this question, we actually have to look at the following questions:

1. Did Consumers like the new flavours?
2. How do they compare with other flavours?
3. Are there any patterns as to why they liked / disliked them?

Secondary Mission

Do individuals like our range and what we have to offer?

To dive deeper into the customer feedback analysis, we can further answer the following:

1. Are reviews that are part of promotions biased?
2. Does the packaging influence customer satisfaction?
3. Do people like our vegan range in comparison to our regular flavours?

Primary Mission

All analyses were conducted using Python and I started by inspecting the dataset:

a. few missing data for product price (hometesterclub and Waitrose):

- in order to fill in the missing values, I researched hometesterclub website and decided to assign 0 as the price (individuals didn't buy the products, they offered reviews)
- for Waitrose, I researched the price on their website and similar ones (Tesco, Sainsbury's) in order to assign a rather accurate value

b. few missing title reviews

- for this, I took the first 3 words from the actual review (review_body) and assigned those as the title

```
# replacing all missing values for hometesterclub with 0 since the retailer is specifically dedicated for reviews
data.loc[data['retailer'].str.lower() == 'hometesterclub', 'product_price'] = 0



# showing the rows where there is missing data
data[data['product_price'].isna()][['product_name', 'retailer', 'product_price']]

# filling in the remaining missing values
data.loc[data['product_name'] == 'Magnum Classic', 'product_price'] = 3.5
data.loc[data['product_name'] == 'Magnum Minis Cookie Mania Ice Cream Sticks', 'product_price'] = 5
data.loc[data['product_name'] == 'Magnum Almond Ice Cream Stick', 'product_price'] = 2.5
data.loc[data['product_name'] == 'Magnum Intense Dark', 'product_price'] = 3.5
data.loc[data['product_name'] == 'Magnum White Chocolate Ice Cream Stick', 'product_price'] = 2.5
data.loc[data['product_name'] == 'Magnum Double Gold Caramel Billionaire Ice Cream Sticks', 'product_price'] = 5
data.loc[data['product_name'] == 'Magnum Vegan Raspberry Swirl', 'product_price'] = 2.5
data.loc[data['product_name'] == 'Magnum Salted Caramel', 'product_price'] = 3.5
data.loc[data['product_name'] == 'Magnum Mint Ice Cream Sticks', 'product_price'] = 3.5
data.loc[data['product_name'] == 'Magnum Vegan Sea Salt Caramel Ice Creams', 'product_price'] = 4.45

data.loc[data['review_title'].isna(), 'review_title'] = data['review_body'].str.split().str[:3].str.join('')
```

Primary Mission

Did consumer like the new flavours?

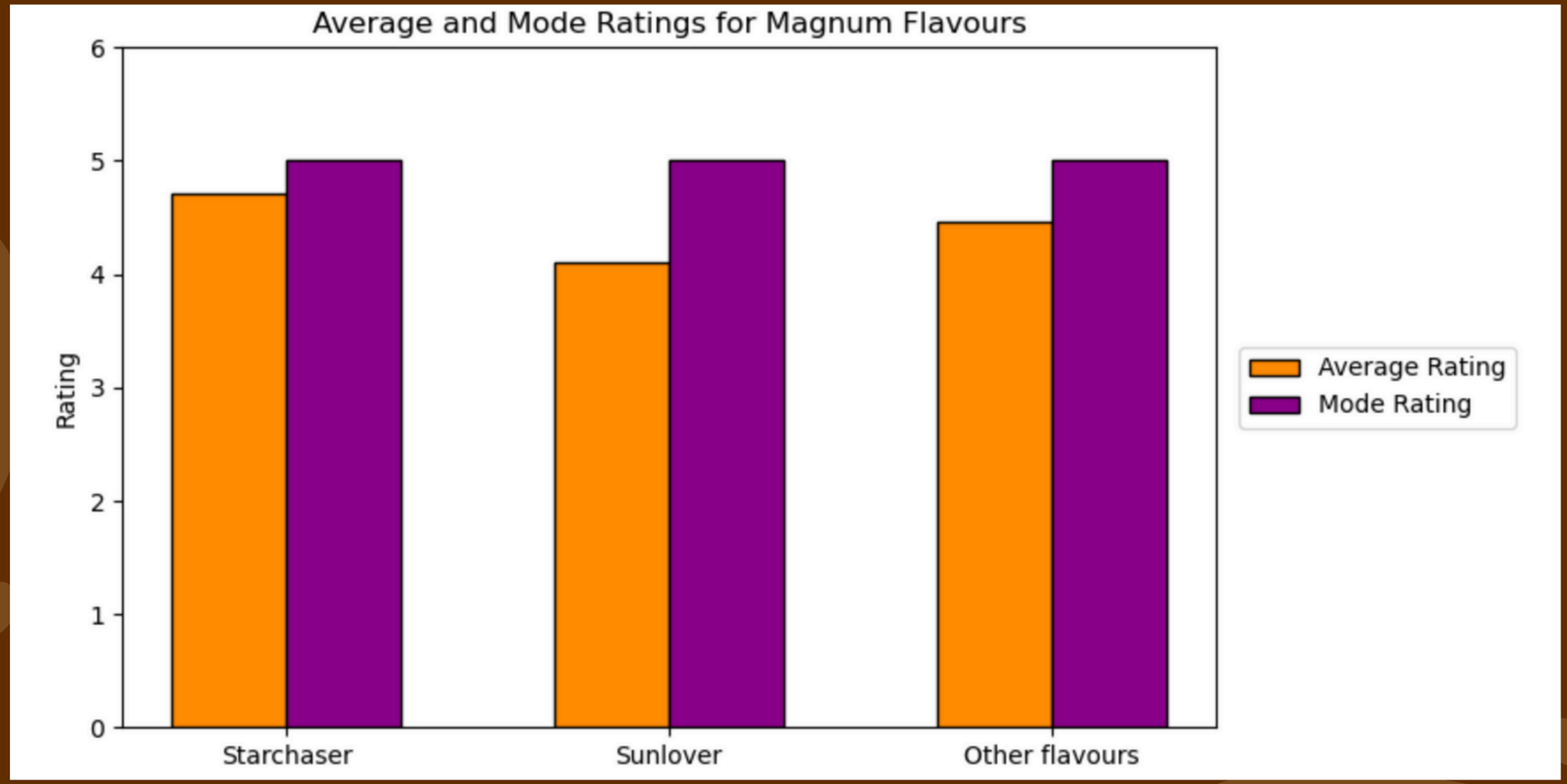
Here we mainly focused on Starchaser vs Sunlover flavours – created a new dataset to only include the rows for both flavours:

Calculated the following:

- mean and mode for Starchaser
- mean and mode for Sunlover
- mean and mode for all remaining flavours

```
# Are you a Sunrise or Sunset ice cream lover?
# Perform a "consumer feedback analysis" on the Starchaser and Sunlover flavours:

# 1. Did consumers like the new flavours?
# let's start by creating a new dataset to include just the rows for starchaser and sunlover ice creams
sunrise_sunset = data[data['product_name'].str.contains('Sunlover', case = False, na = False) | data['product_name'].str.contains(
    'Starchaser', case = False, na = False)]
sunrise_sunset

# we'll calculate the average and mode for each falvour
starchaser_rating = sunrise_sunset[sunrise_sunset['product_name'].str.contains('Starchaser', case = False)]['review_star_rating']
starchaser_mean = starchaser_rating.mean()
starchaser_mode = starchaser_rating.mode()[0]
print(f'Starchaser - Average rating: {starchaser_mean: .2f}, Mode rating: {starchaser_mode}')

sunlover_rating = sunrise_sunset[sunrise_sunset['product_name'].str.contains('Sunlover', case = False)]['review_star_rating']
sunlover_mean = sunlover_rating.mean()
sunlover_mode = sunlover_rating.mode()[0]
print(f'Sunlover - Average rating: {sunlover_mean: .2f}, Mode rating: {sunlover_mode}')

other_flavours = data[~data['product_name'].str.contains('Starchaser | Sunlover', case = False)]
other_mean = other_flavours['review_star_rating'].mean()
other_mode = other_flavours['review_star_rating'].mode()[0]
print(f'Other flavour - Average rating: {other_mean: .2f}, Mode rating: {other_mode}')

flavours = ['Starchaser', 'Sunlover', 'Other flavours']
mean_ratings = [starchaser_mean, sunlover_mean, other_mean]
mode_ratings = [starchaser_mode, sunlover_mode, other_mode]
x = range(len(flavours))

plt.figure(figsize = (8,5))
plt.bar([i-0.15 for i in x], mean_ratings, width = 0.3, label = 'Average Rating', color = 'darkorange', edgecolor = 'black')
plt.bar([i+0.15 for i in x], mode_ratings, width = 0.3, label = 'Mode Rating', color = 'darkmagenta', edgecolor = 'black')
plt.xticks(x, flavours)
plt.ylabel('Rating')
plt.ylim(0,6)
plt.title ('Average and Mode Ratings for Magnum Flavours')
plt.legend(loc = 'center left', bbox_to_anchor = (1.01, 0.5))
plt.savefig('starchaser_sunlover_vs_others.png', dpi = 300, bbox_inches = 'tight')
plt.show()
```

Results



Notes:

- customers prefer Starchaser (registering a mean of 4.71 in ratings), followed by all other flavours (4.46), and lastly Sunlover (4.11)
- the majority of ratings given was 5 across all flavours

Primary Mission

How do they compare with other flavours?

To compare with other flavours, I decided to chose from rather classic but favored choices:

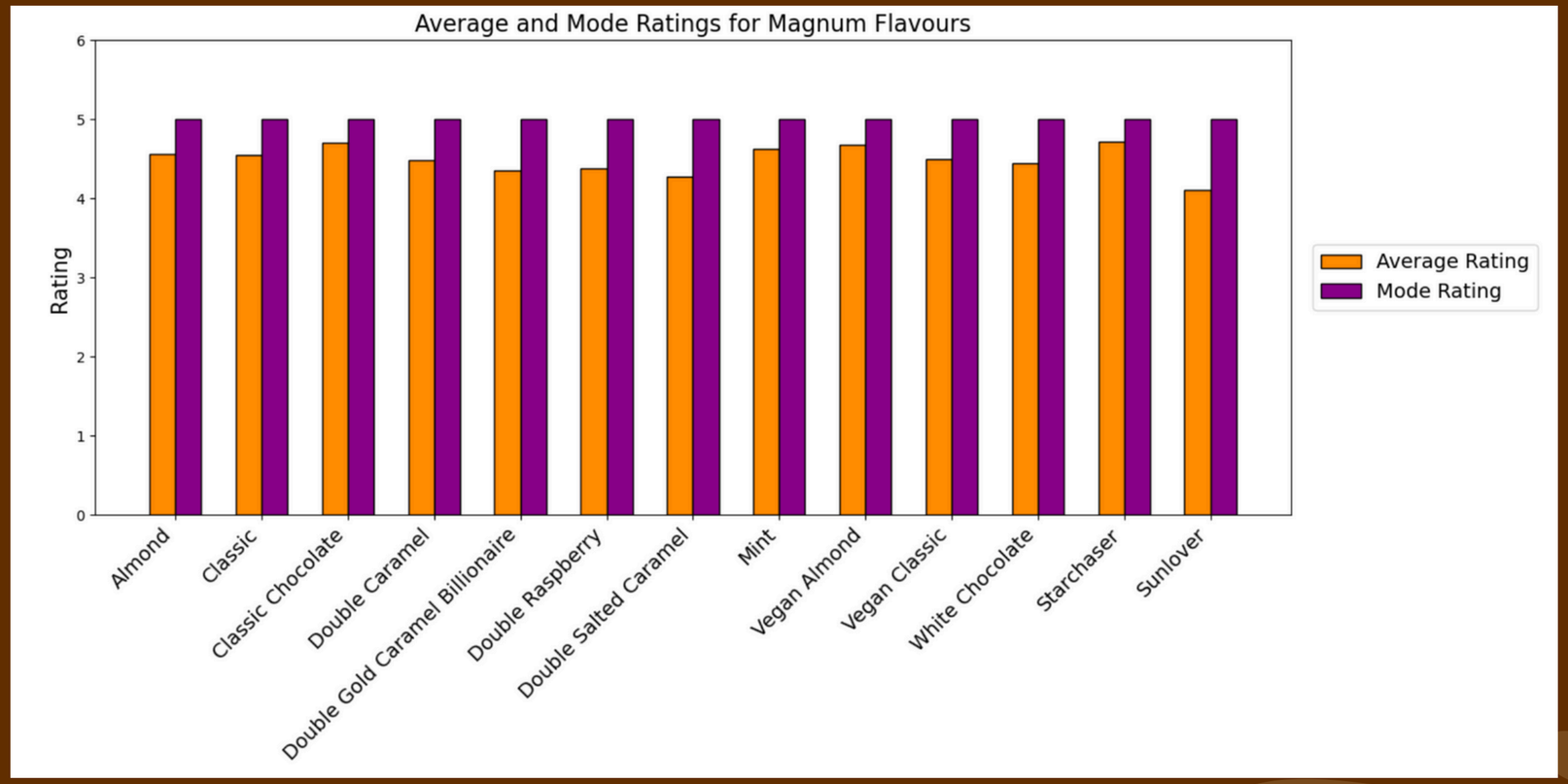
- white chocolate
- double salted caramel
- vegan classic
- vegan almond
- double gold caramel billionaire
- almond
- mint
- classic chocolate
- classic
- double caramel
- double raspberry

I calculated the means and modes for each flavour, then compared against Starchaser and Sunlover.

This way, we dive deeper into customers' preferences and we can understand them better.

```python
# 2. How do they compare with other flavours?
comparison_keywords = ['white chocolate', 'double salted caramel', 'vegan classic', 'vegan almond',
                       'double gold caramel billionaire', 'almond', 'mint', 'classic chocolate', 'classic', 'double caramel',
                      'double raspberry']
def map_keyword(product_name):
    for kw in comparison_keywords:
        if kw in product_name.lower():
            return kw.title()
    return None

comparison_data = comparison_data.copy()
comparison_data['comparison_keyword'] = comparison_data['product_name'].apply(map_keyword)
rating_summary = comparison_data.groupby('comparison_keyword')['review_star_rating'].agg(['mean', lambda x: x.mode()[0]])
rating_summary.rename(columns = {'mean': 'Average Rating', '<lambda_0>': 'Mode Rating'}, inplace = True)
print(rating_summary, '\n')

#plot for all flavours
flavours = list(rating_summary.index) + ['Starchaser', 'Sunlover']
mean_ratings = list(rating_summary['Average Rating']) + [starchaser_mean, sunlover_mean]
mode_ratings = list(rating_summary['Mode Rating']) + [starchaser_mode, sunlover_mode]
x = range(len(flavours))
plt.figure(figsize = (15, 6))
plt.bar([i-0.15 for i in x], mean_ratings, width = 0.3, label = 'Average Rating', color = 'darkorange', edgecolor = 'black')
plt.bar([i+0.15 for i in x], mode_ratings, width = 0.3, label = 'Mode Rating', color = 'darkmagenta', edgecolor = 'black')
plt.xticks(x, flavours, rotation = 45, ha = 'right', fontsize = 14)
plt.ylabel('Rating', fontsize = 15)
plt.ylim(0,6)
plt.title ('Average and Mode Ratings for Magnum Flavours', fontsize = 16)
plt.legend(loc = 'center left', bbox_to_anchor = (1.01, 0.5), fontsize = 14)
#plt.tight_layout()
plt.savefig('ss_comparison_with_other_flavours.png', dpi = 300, bbox_inches = 'tight')
plt.show()
```

Results



Notes:

- Classic Chocolate proved to be all-time favoured (4.7) , closely followed by Vegan Almond (4.67) and Mint (4.63) and close to the new Starchaser flavour
- on the other hand, Sunlover proved to be the least rated among all flavours

Primary Mission

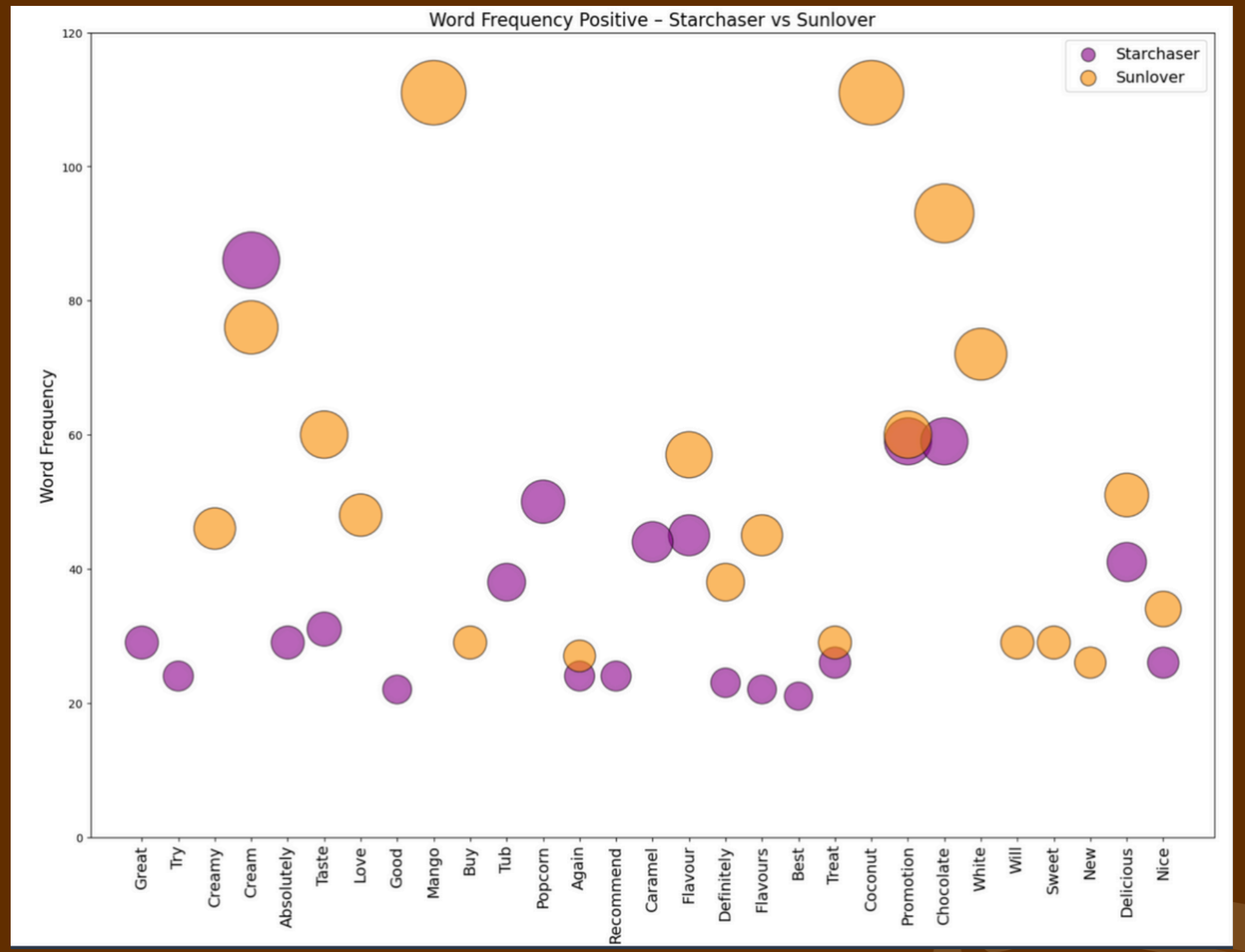
Why did customers like the new flavours?

I created a list that includes the top 20 words, taken from their reviews as to describe the reasons for liking Starchaser and Sunlover

- I did exclude a number of generic words (stopwords) that do not significantly impact the analysis
- the results are presented as a bubble chart

```
# 3. Are there any patterns as to why they liked them?

from collections import Counter
import re

def get_top_words(review_body, top_n = 20):
    all_text = ' '.join(review_body).lower()
    words = re.findall(r'\b\w+\b', all_text)
    stopwords = set(['the', 'and', 'a', 'is', 'it', 'i', 'to', 'of', 'in', 'for', 'on', 'with', 'this', 'that', 'my',
                     'as', 'was', 'but', 'so', 'at', 'had', 'they', 'from', 'you', 'ice', 'are', 'not', 'these', 'review', 'them',
                     'really', 'very', 'like', 'magnum', 'have', 'be', 'would', 'just', 'âôt', 'too', 's', 'part', 'collected',
                     'one'])
    filtered_words = [w for w in words if w not in stopwords]
    word_counts = Counter(filtered_words)
    return word_counts.most_common(top_n)

starchaser_reviews = sunrise_sunset[sunrise_sunset['product_name'].str.contains('Starchaser', case = False)]['review_body']
starchaser_top_words = get_top_words(starchaser_reviews)
print('Starchaser top words:', starchaser_top_words, '\n')

sunlover_reviews = sunrise_sunset[sunrise_sunset['product_name'].str.contains('Sunlover', case = False)]['review_body']
sunlover_top_words = get_top_words(sunlover_reviews)
print('Sunlover top words:', sunlover_top_words)
print('\n')

starchaser_words, starchaser_counts = zip (*starchaser_top_words)
sunlover_words, sunlover_counts = zip (*sunlover_top_words)
all_words = list(set(starchaser_words) | set(sunlover_words))
word_pos = {word: i for i, word in enumerate(all_words)}

x_starchaser = [word_pos[w] for w in starchaser_words]
y_starchaser = starchaser_counts
size_starchaser = [c * 30 for c in starchaser_counts]

x_sunlover = [word_pos[w] for w in sunlover_words]
y_sunlover = sunlover_counts
size_sunlover = [c * 30 for c in sunlover_counts]

plt.figure(figsize = (18,13))
plt.scatter(x_starchaser, y_starchaser, s = size_starchaser, color = 'darkmagenta', alpha = 0.6, label = 'Starchaser', edgecolors = 'black')
plt.scatter(x_sunlover, y_sunlover, s = size_sunlover, color = 'darkorange', alpha = 0.6, label = 'Sunlover', edgecolors = 'black')
capitalized_words = [w.title() for w in all_words]
plt.xticks(range(len(all_words)), capitalized_words, rotation = 90, fontsize = 14)
plt.ylabel('Word Frequency', fontsize = 15)
plt.ylim(0, 120)
plt.title('Word Frequency Positive – Starchaser vs Sunlover', fontsize = 16)
plt.legend(markerscale = 0.3, fontsize = 14)
plt.savefig('bubble_chart_pos_frequency.png', dpi = 300, bbox_inches = 'tight')
plt.show()
```

Results



Notes:

- for Starchaser, the top words were flavour-related (caramel, popcorn, chocolate), describing it as delicious and creamy, with customers further recommending the flavour
- for Sunlover, we can notice a similar trend, with flavours (mango, chocolate, coconut), customers loving it and willing to re-purchase this flavour

Primary Mission

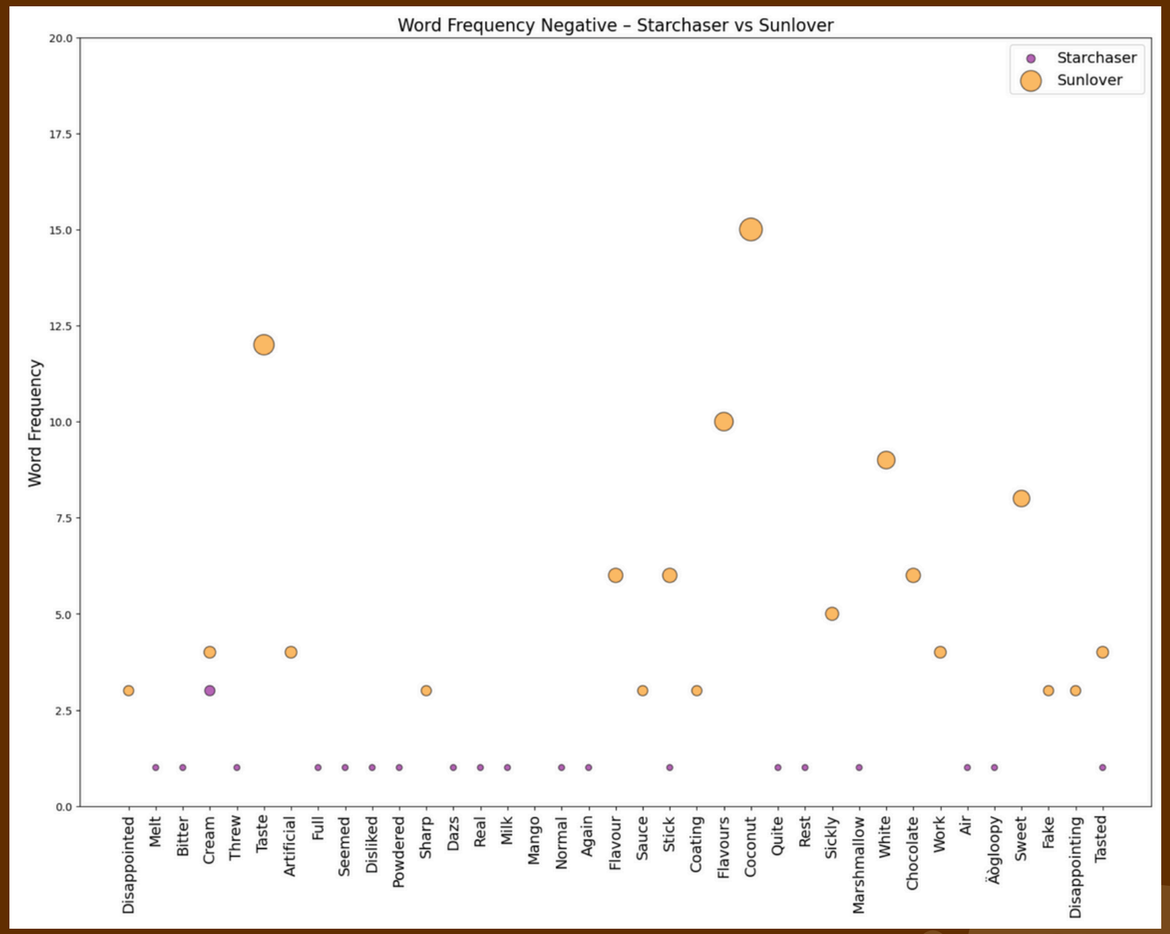
Why did customers dislike the new flavours?

I applied the same rationale as in the previous analysis, where I looked at reasons for liking the new flavours, but here I investigated the reasons for disliking the new flavours

- looked at the top 20 words to describe dissatisfaction among customers

```python
# why they disliked them?

def get_top_words(review_body, top_n = 20):
    all_text = ' '.join(review_body).lower()
    words = re.findall(r'\b\w+\b', all_text)
    stopwords = set(['the', 'and', 'a', 'is', 'it', 'i', 'to', 'of', 'in', 'for', 'on', 'with', 'this', 'that', 'my',
                     'as', 'was', 'but', 'so', 'at', 'had', 'they', 'from', 'you', 'ice', 'are', 'not', 'these', 'review', 'them',
                     'really', 'very', 'like', 'magnum', 'have', 'be', 'would', 'just', 'äôt', 'too', 's', 'part', 'collected',
                     'one', 'away', 'love', 'only', 'h', 'together', 'im', 'should', 'however', 'something', 'product',
                     'agen', 'nice', 'great', 'no', 'all', 't', 'both', 'bit', 'don', 'go', 'day', 'using', 'does',
                     'next', 'must', 'never'])
    filtered_words = [w for w in words if w not in stopwords]
    word_counts = Counter(filtered_words)
    return word_counts.most_common(top_n)

negative_threshold = 2
starchaser_disliked_reviews = sunrise_sunset[(sunrise_sunset['product_name'].str.contains('Starchaser', case = False, na = False)) &
    (sunrise_sunset['review_star_rating'] <= negative_threshold)]['review_body'].dropna()
sunlover_disliked_reviews = sunrise_sunset[(sunrise_sunset['product_name'].str.contains('Sunlover', case = False, na = False)) &
    (sunrise_sunset['review_star_rating'] <= negative_threshold)]['review_body'].dropna()

def safe_get_top_words(series, top_n = 20):
    if series.empty:
        return []
    return get_top_words(series, top_n = top_n)

starchaser_top_disliked = safe_get_top_words(starchaser_disliked_reviews, top_n = 20)
sunlover_top_disliked = safe_get_top_words(sunlover_disliked_reviews, top_n = 20)

if starchaser_top_disliked:
    starchaser_words, starchaser_counts = zip(*starchaser_top_disliked)
else:
    starchaser_words, starchaser_counts = [],[]

if sunlover_top_disliked:
    sunlover_words, sunlover_counts = zip(*sunlover_top_disliked)
else:
    sunlover_words, sunlover_counts = [],[]

all_words = list(set(starchaser_words) | set(sunlover_words))
word_pos = {word: i for i, word in enumerate(all_words)}

print("Starchaser top disliked words:", starchaser_top_disliked, '\n')
print("Sunlover top disliked words:", sunlover_top_disliked, '\n')

x_starchaser = [word_pos[w] for w in starchaser_words]
y_starchaser = starchaser_counts
size_starchaser = [c * 30 for c in starchaser_counts]

x_sunlover = [word_pos[w] for w in sunlover_words]
y_sunlover = sunlover_counts
size_sunlover = [c * 30 for c in sunlover_counts]

plt.figure(figsize = (18,13))
plt.scatter(x_starchaser, y_starchaser, s = size_starchaser, color = 'darkmagenta', alpha = 0.6, label = 'Starchaser', edgecolors = 'black')
plt.scatter(x_sunlover, y_sunlover, s = size_sunlover, color = 'darkorange', alpha = 0.6, label = 'Sunlover', edgecolors = 'black')
capitalized_words = [w.title() for w in all_words]
plt.xticks(range(len(all_words)), capitalized_words, rotation = 90, fontsize = 14)
plt.ylabel('Word Frequency', fontsize = 15)
plt.ylim(0, 20)
plt.title('Word Frequency Negative - Starchaser vs Sunlover', fontsize = 16)
plt.legend(markerscale = 1, fontsize = 14)
plt.savefig('bubble_chart_neg_frequency.png', dpi = 300, bbox_inches = 'tight')
plt.show()
```

Results



Notes:

- a feeling of disappointment can be noticed for both ice creams, due to an artificial, powder-like, sticky, bitter taste
- some delivery / packaging issues may have risen: ice cream melted, the packaging was not full
- a similarity in taste to the Häagen-Dazs ice cream
- some customers also voiced that they wouldn't buy the ice cream a second time

Secondary Mission

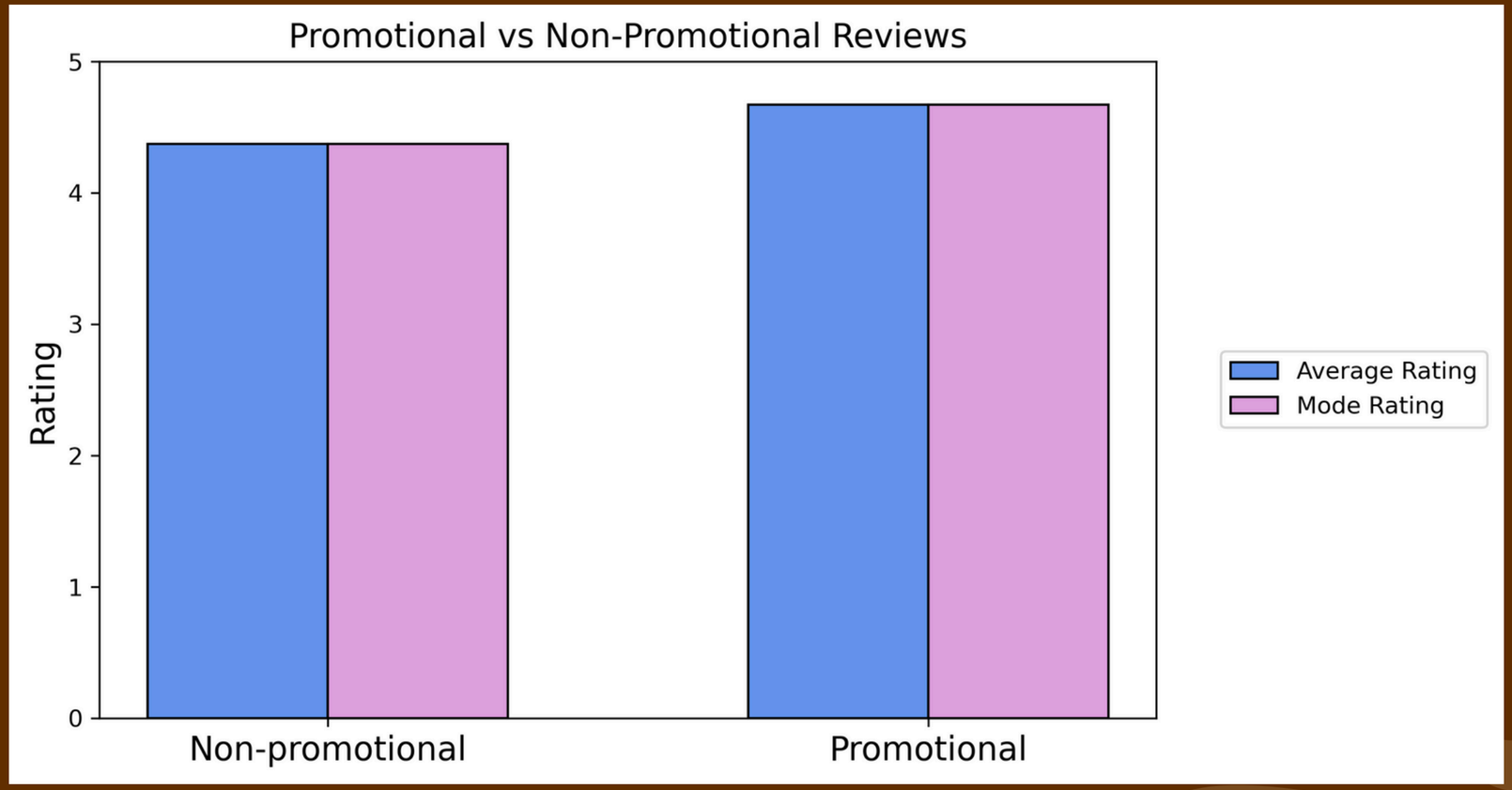
Were promotional reviews biased?

I created a new dataset in which a new column was added (promo_group), that informed us whether the review was part of a promotional campaign or not

- then calculated the mean and mode for 'promotional' reviews vs 'non-promotional' reviews

```
# Secondary mission
#1. promotional vs normal reviews – are reviews part of the promotional campaign biased? – general

promo_data = data.copy()
promo_data['promo_group'] = promo_data['promotional_review'].apply(lambda x: 'Promotional' if x else 'Non-promotional')
promo_summary = promo_data.groupby('promo_group')['review_star_rating'].agg(mean_rating = 'mean',
                                                                             mode_rating = lambda x: x.mode().iloc[0]).reset_index()
print(promo_summary, '\n')

labels = promo_summary['promo_group']
means = promo_summary['mean_rating']
modes = promo_summary['mode_rating']

x = range(len(labels))
plt.figure(figsize = (8,5))
plt.bar([i-0.15 for i in x], means, width = 0.3, label = 'Average Rating', color = 'cornflowerblue', edgecolor = 'black')
plt.bar([i+0.15 for i in x], means, width = 0.3, label = 'Mode Rating', color = 'plum', edgecolor = 'black')
plt.xticks(x, labels, fontsize = 14)
plt.ylabel('Rating', fontsize = 14)
plt.title('Promotional vs Non-Promotional Reviews', fontsize = 14)
plt.ylim(0,5)
plt.legend(fontsize = 10, loc = 'center left', bbox_to_anchor = (1.05, 0.5))
plt.savefig('promotional_vs_nonp.png', dpi = 300, bbox_inches = 'tight')
plt.show()
```

Results



Notes:

- there is a tendency for promotional reviews to be biased (4.67), as both the mean and mode for ratings were slightly higher than for those that were non-promotional (4.37)

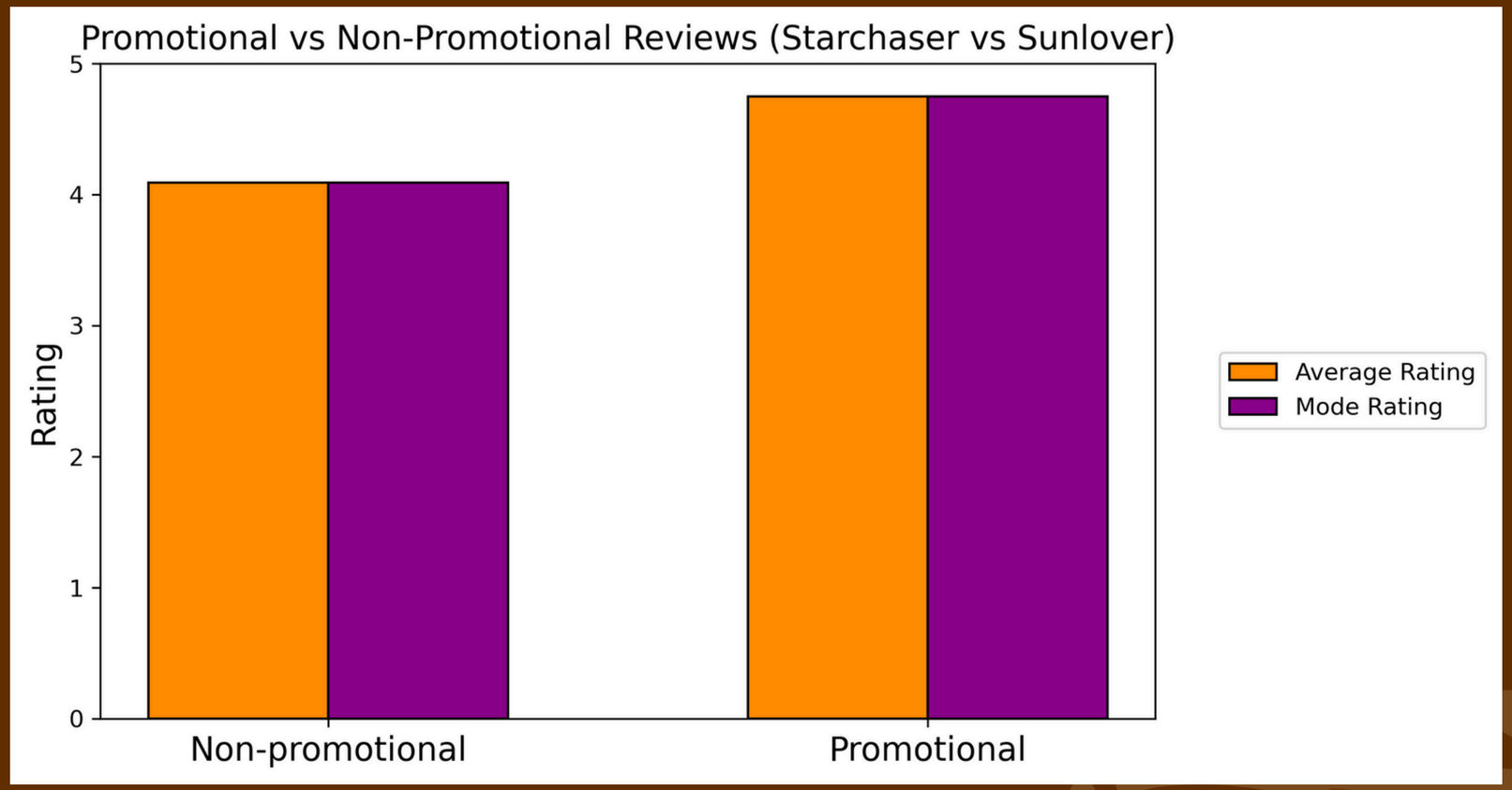
Secondary Mission

Were promotional reviews biased for Starchaser and Sunlover?

Applying the same mechanisms as before, I was also curious to investigate whether the reviews for the new flavour were biased in any way

```
# Secondary mission
#1. promotional vs normal reviews – are reviews part of the promotional campaign biased? – starchaser vs sunlover

promo_data_ss = sunrise_sunset.copy()
promo_data_ss['promo_group'] = promo_data_ss['promotional_review'].apply(lambda x: 'Promotional' if x else 'Non-promotional')
promo_summary_ss = promo_data_ss.groupby('promo_group')['review_star_rating'].agg(mean_rating = 'mean',
                                                                                   mode_rating = lambda x: x.mode().iloc[0]).reset_index()
print(promo_summary_ss, '\n')

labels = promo_summary_ss['promo_group']
means = promo_summary_ss['mean_rating']
modes = promo_summary_ss['mode_rating']

x = range(len(labels))
plt.figure(figsize = (8,5))
plt.bar([i-0.15 for i in x], means, width = 0.3, label = 'Average Rating', color = 'darkorange', edgecolor = 'black')
plt.bar([i+0.15 for i in x], means, width = 0.3, label = 'Mode Rating', color = 'darkmagenta', edgecolor = 'black')
plt.xticks(x, labels, fontsize = 14)
plt.ylabel('Rating', fontsize = 14)
plt.title('Promotional vs Non-Promotional Reviews (Starchaser vs Sunlover)', fontsize = 14)
plt.ylim(0,5)
plt.legend(fontsize = 10, loc = 'center left', bbox_to_anchor = (1.05, 0.5))
plt.savefig('promotional_vs_nonp_ss.png', dpi = 300, bbox_inches = 'tight')
plt.show()
```

Results



Notes:

- results revealed a similar trend as in the previous analysis, but with a sharper contrast than before, with promotional reviews rating the new flavours much higher (4.75), than non-promotional reviews (4.1)

Secondary Mission

What is the most advantageous packaging?

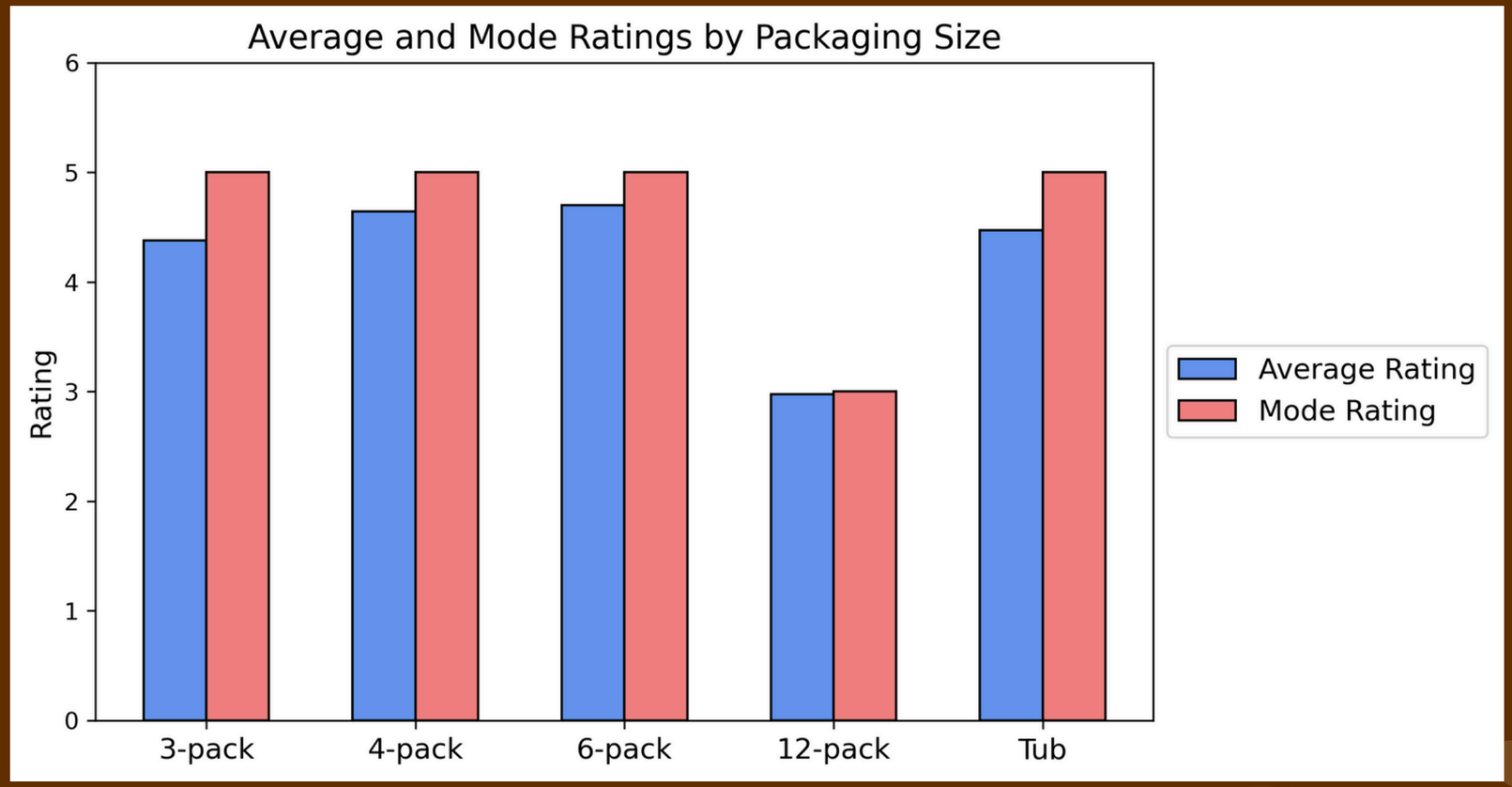
I investigated the dataset in order to categorise the packaging options available:

- 3-pack
- 4-pack
- 6-pack
- 12-pack
- tub

To the best packaging option, we referenced the reviews once again, but we also kept count of the amount of ice cream that was bought according to each category

```
def extract_pack_size(name):
    name = str(name).lower()

    # 1. Match '3x100ml' format
    match1 = re.search(r'(\d+)\s*x\s*\d+\s*ml', name)
    if match1:
        return int(match1.group(1))

    # 2. Match explicit 'pack' format
    match2 = re.search(r'(\d+)\s*pack', name)
    if match2:
        return int(match2.group(1))

    # 3. Match number before 'mini' or 'ice cream' for things like "6 Mini Double Caramel..."
    match3 = re.search(r'\b(\d+)\b.*?(?:mini|ice cream)s?', name, flags = re.IGNORECASE)
    if match3:
        return int(match3.group(1))

    # 4. Single tubs (with ml but no pack info)
    if 'tub' in name or re.search(r'\b\d+\s*ml\b', name):
        return 'Tub'

    return np.nan

data['pack_size'] = data['product_name']. apply(extract_pack_size)
pack_data = data[data['pack_size'].isin([3,4,6,12, 'Tub'])].copy()
pack_summary = pack_data.groupby('pack_size')['review_star_rating'].agg(
    mean_rating = 'mean',
    mode_rating = lambda x: x.mode().iloc[0] if not x.mode().empty else np.nan,
    count = 'size').reset_index()
print(pack_summary, '\n')

x = np.arange(len(pack_summary['pack_size']))
width = 0.3

plt.figure(figsize = (8,5))
plt.bar([i - width / 2 for i in x], pack_summary['mean_rating'], width = width, color = 'cornflowerblue', edgecolor = 'black',
        label = 'Average Rating')
plt.bar([i + width / 2 for i in x], pack_summary['mode_rating'], width = width, color = 'lightcoral', edgecolor = 'black',
        label = 'Mode Rating')
plt.xticks(x, [f"{p}-pack" if isinstance(p, (int, np.integer)) else str(p)
               for p in pack_summary['pack_size']], fontsize = 12)
plt.ylabel('Rating', fontsize = 12)
plt.ylim(0,6)
plt.title('Average and Mode Ratings by Packaging Size', fontsize = 14)
plt.legend(fontsize = 12, loc = 'center left', bbox_to_anchor = (1, 0.5))
plt.savefig('packaging_all.png', dpi = 300, bbox_inches = 'tight')
plt.show()
```

Results



Notes:

- according to the reviews, the 6-pack option was rated highest (4.7) and a mode of 5, whereas the lowest one was represented by the 12-pack (2.98, mode 3)
- on the other hand, the 3-pack option was the most bought ones (1652 times), followed by tub (1518), 4-pack (1156), 6-pack (911) and lastly, the 12-pack (90)

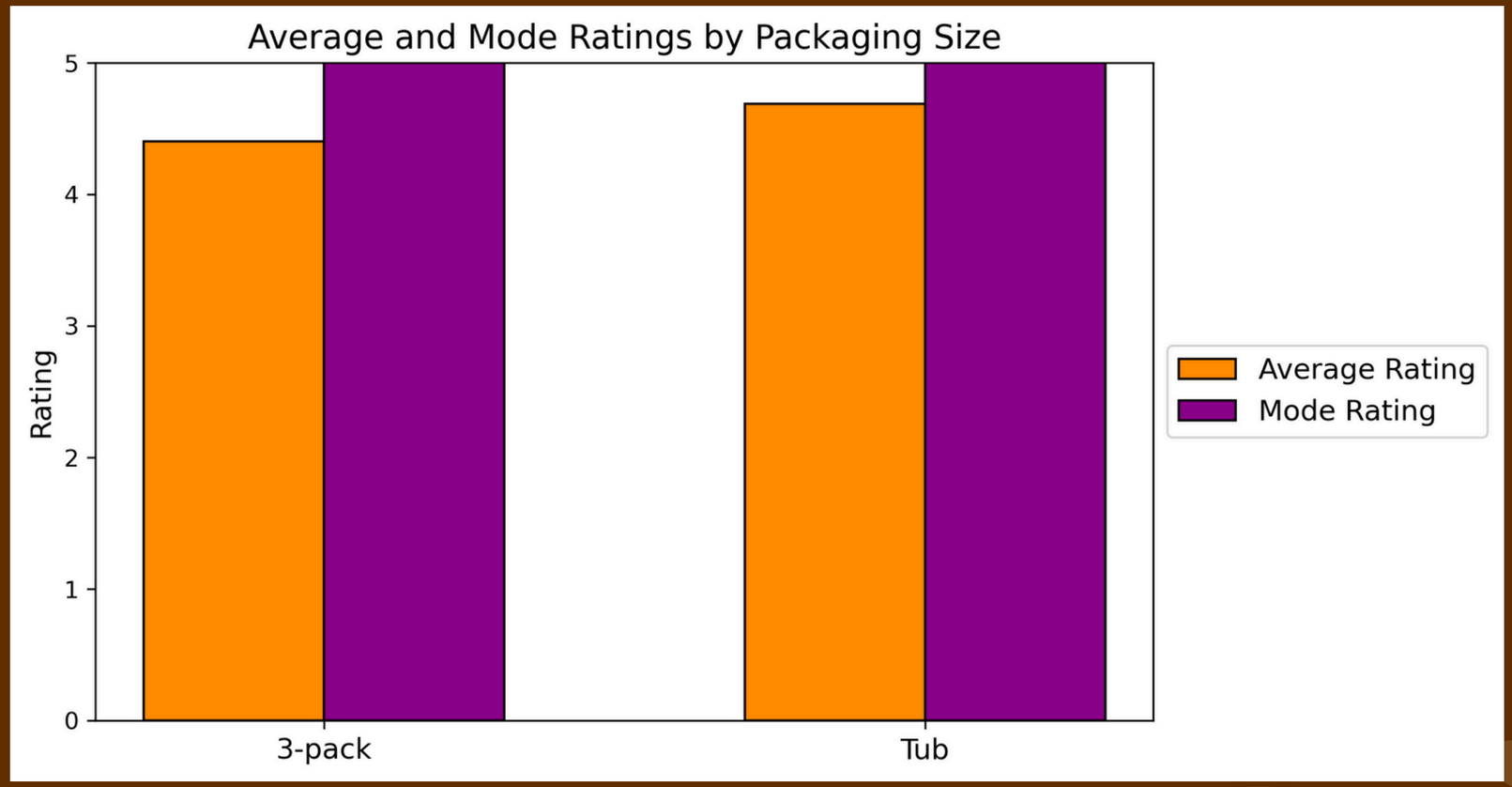
Secondary Mission

What is the most advantageous packaging?

The same rationale was applied when investigating the same question for Starchaser vs Sunlover

```
def extract_pack_size(name):
    name = str(name).lower()

    # 1. Match '3x100ml' format
    match1 = re.search(r'(\d+)\s*x\s*\d+\s*ml', name)
    if match1:
        return int(match1.group(1))

    # 2. Match explicit 'pack' format
    match2 = re.search(r'(\d+)\s*pack', name)
    if match2:
        return int(match2.group(1))

    # 3. Match number before 'mini' or 'ice cream' for things like "6 Mini Double Caramel..."
    match3 = re.search(r'\b(\d+)\b.*?(?:mini|ice cream)s?', name, flags = re.IGNORECASE)
    if match3:
        return int(match3.group(1))

    # 4. Single tubs (with ml but no pack info)
    if 'tub' in name or re.search(r'\b\d+\s*ml\b', name):
        return 'Tub'

    return np.nan

ss_pack_data = sunrise_sunset.copy()
ss_pack_data['pack_size'] = ss_pack_data['product_name']. apply(extract_pack_size)
ss_pack_data = ss_pack_data[ss_pack_data['pack_size'].isin([3,4,6,12, 'Tub'])].copy()
ss_pack_summary = ss_pack_data.groupby('pack_size')['review_star_rating'].agg(
    mean_rating = 'mean',
    mode_rating = lambda x: x.mode().iloc[0] if not x.mode().empty else np.nan,
    count = 'size').reset_index()
print(ss_pack_summary, '\n')

x = np.arange(len(ss_pack_summary['pack_size']))
width = 0.3

plt.figure(figsize = (8,5))
plt.bar([i - width / 2 for i in x], ss_pack_summary['mean_rating'], width = width, color = 'darkorange', edgecolor = 'black',
        label = 'Average Rating')
plt.bar([i + width / 2 for i in x], ss_pack_summary['mode_rating'], width = width, color = 'darkmagenta', edgecolor = 'black',
        label = 'Mode Rating')
plt.xticks(x, [f"{p}-pack" if isinstance(p, (int, np.integer)) else str(p)
               for p in ss_pack_summary['pack_size']], fontsize = 12)
plt.ylabel('Rating', fontsize = 12)
plt.ylim(0,5)
plt.title('Average and Mode Ratings by Packaging Size', fontsize = 14)
plt.legend(fontsize = 12, loc = 'center left', bbox_to_anchor = (1, 0.5))
plt.savefig('packaging_ss.png', dpi = 300, bbox_inches = 'tight')
plt.show()
```

Results



Notes:

- for the 2 new flavours, we can see that they only come in two options – 3-pack and tub
- the tub version seemed to have been rated higher (4.69) compared to the 3-pack (4.4), with the tub option being bought more as well (106 vs 100)

Secondary Mission

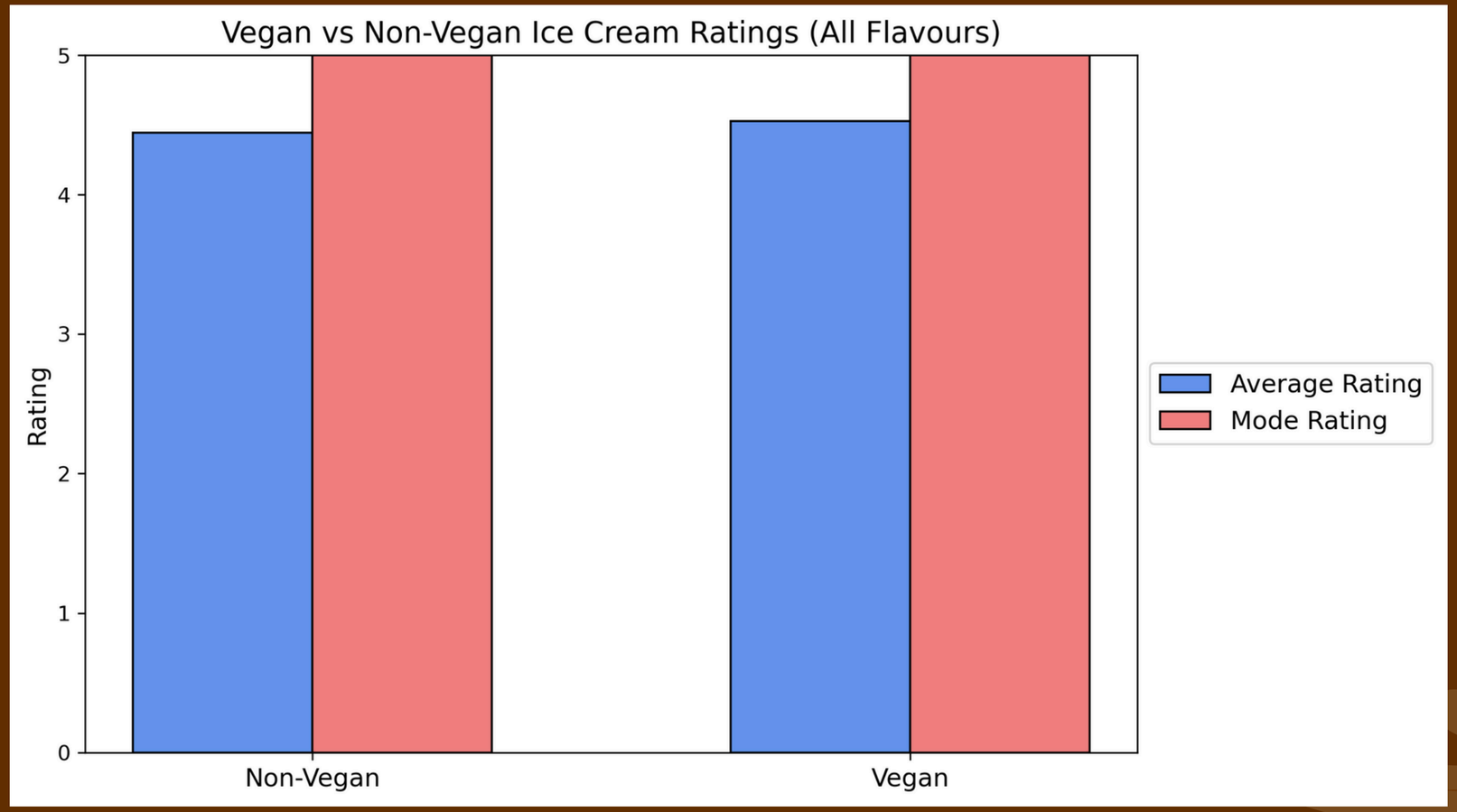
Is the vegan or the non-vegan option better?

In order to tell which option (vegan or non-vegan) was preferred by customers, I created an additional column (vegan_group) to the dataset and categorise each ice cream entry based on whether the product_name included vegan or not.

- this was done across all flavours, as Starchaser and Sunlover don't have a vegan option

```
data['vegan_group'] = data['product_name'].str.contains('vegan', case = False, na = False)
data['vegan_group'] = data['vegan_group'].apply(lambda x: 'Vegan' if x else 'Non-Vegan')

vegan_summary = data.groupby('vegan_group')['review_star_rating'].agg(mean_rating = 'mean',
                                                                      mode_rating = lambda x: x.mode().iloc[0]
                                                                      if not x.mode().empty else np.nan,
                                                                     count = 'size').reset_index()
print(vegan_summary, '\n')

x = np.arange(len(vegan_summary))
width = 0.3

plt.figure(figsize = (9,6))
plt.bar(x - width / 2, vegan_summary['mean_rating'], width = width, color = 'cornflowerblue', edgecolor = 'black', label = 'Average Rating')
plt.bar(x + width / 2, vegan_summary['mode_rating'], width = width, color = 'lightcoral', edgecolor = 'black', label = 'Mode Rating')
plt.xticks(x, vegan_summary['vegan_group'], fontsize = 12)
plt.ylabel('Rating', fontsize = 12)
plt.ylim(0,5)
plt.title('Vegan vs Non-Vegan Ice Cream Ratings (All Flavours)', fontsize = 14)
plt.legend(fontsize = 12, loc = 'center left', bbox_to_anchor = (1, 0.5))
plt.savefig('vegan_vs_nonvegan.png', dpi = 300, bbox_inches = 'tight')
plt.show()
```

Results



Notes:

- we can observe that the vegan option was rated slightly higher among consumers (4.44) compared to the non-vegan option (4.53)
- however, the non-vegan option was slightly preferred (7904) than the vegan one (1263) as it was bought more times than its alternative option

Final Verdict

Across the analyses ran, we can conclude the following:

- Starchaser was the preferred flavour among all examined flavours
- promotional campaigns should persist as they bring attention to new product and the ratings are in the company's favour
- the 3-pack and tub options were the most commonly bought packaging options
- more vegan options should be put out in the market, as the ratings were very similar to the non-vegan option (clearly preferred by customers in terms of sales quantity)