In [1]:

```python
import pandas as pd
from scipy.stats import ttest_rel, wilcoxon
from scipy.stats import pearsonr
import numpy as np
import statsmodels.api as sm
from statsmodels.formula.api import ols

# Reading in the data
gzero_data = 'Gatorade_Zero_Taste_Test_Survey_Responses_2.csv'
rawdata = pd.read_csv(gzero_data)

# Separating the data for each product
original = rawdata[rawdata['Product'] == 'Original']
new_formulation = rawdata[rawdata['Product'] == 'New Formulation']

# Ensuring the data is aligned by Panelist_ID
original = original.sort_values('Panelist_ID')
new_formulation = new_formulation.sort_values('Panelist_ID')

# Performing paired t-test for each liking attribute
liking_attributes = [col for col in original.columns if 'Liking' in col]
t_test_results = {}
wilcoxon_test_results = {}

for attribute in liking_attributes:
    t_stat, p_value = ttest_rel(original[attribute], new_formulation[attribute])
    t_test_results[attribute] = (t_stat, p_value)

# Performing Wilcoxon signed-rank test to doublecheck results
    w_stat, p_value = wilcoxon(original[attribute], new_formulation[attribute])
    wilcoxon_test_results[attribute] = (w_stat, p_value)

# Displaying results
t_test_results_df = pd.DataFrame.from_dict(t_test_results, orient='index', colum
wilcoxon_test_results_df = pd.DataFrame.from_dict(wilcoxon_test_results, orient=

print("Paired t-test Results:")
print(t_test_results_df)

print("\nWilcoxon Signed-Rank Test Results:")
print(wilcoxon_test_results_df)


#Printing mean scores
liking_means = rawdata.groupby('Product')[liking_attributes].mean().round(2)

# Transposing the dataframe for better readability
liking_means_transposed = liking_means.T
liking_means_transposed.reset_index(inplace=True)
liking_means_transposed.columns = ['Liking Attribute', 'Original', 'New Formulat

# Displaying the table
print("\nLiking Means:")

print(liking_means_transposed)
```

```
Paired t-test Results:
                        t_stat   p_value
Overall_Liking        0.000000  1.000000
```

```
Flavor_Liking        -1.238540  0.219184
Aroma_Liking          0.454048  0.651039
Color_Liking         -0.136003  0.892165
Orange_Flavor_Liking  0.474017  0.636795
Sweetness_Liking     -0.671139  0.504090
Mouthfeel_Liking     -0.056902  0.954767
Aftertaste_Liking    -1.547787  0.125670

Wilcoxon Signed-Rank Test Results:
                      w_stat   p_value
Overall_Liking        1240.5  0.990636
Flavor_Liking         1086.0  0.199374
Aroma_Liking          1266.5  0.643467
Color_Liking          1285.5  0.872598
Orange_Flavor_Liking  1151.5  0.593412
Sweetness_Liking      1151.5  0.467149
Mouthfeel_Liking      1253.5  0.888011
Aftertaste_Liking     1138.0  0.128262

Liking Means:
       Liking Attribute  Original  New Formulation
0        Overall_Liking      5.29             5.29
1         Flavor_Liking      5.56             4.96
2          Aroma_Liking      5.14             5.36
3          Color_Liking      5.42             5.36
4  Orange_Flavor_Liking      5.05             5.28
5      Sweetness_Liking      5.69             5.39
6      Mouthfeel_Liking      5.31             5.29
7     Aftertaste_Liking      5.60             4.94
```

In [2]:

```
# Now working on JAR scores
# Define the list of JAR attributes
jar_attributes = [col for col in rawdata.columns if 'JAR' in col]

# Function to calculate the JAR category percentages
def calculate_jar_percentages(df, attributes):
    jar_percentages = pd.DataFrame(index=attributes, columns=['Not Enough', 'JAR

    for attribute in attributes:
        not_enough = df[attribute].isin([1, 2]).sum()
        just_about_right = df[attribute].isin([3]).sum()
        too_much = df[attribute].isin([4, 5]).sum()
        total = len(df)

        jar_percentages.at[attribute, 'Not Enough'] = (not_enough / total) * 100
        jar_percentages.at[attribute, 'JAR'] = (just_about_right / total) * 100
        jar_percentages.at[attribute, 'Too Much'] = (too_much / total) * 100

    return jar_percentages.round(2)

# Separate the data for each product
original_data = rawdata[rawdata['Product'] == 'Original']
new_formulation_data = rawdata[rawdata['Product'] == 'New Formulation']

# Calculate the JAR category percentages for each product
original_jar_percentages = calculate_jar_percentages(original_data, jar_attribut
new_formulation_jar_percentages = calculate_jar_percentages(new_formulation_data


original_jar_percentages, new_formulation_jar_percentages
```

Out[2]:

```
(                  Not Enough    JAR Too Much
 Flavor_JAR             31.25   17.5    51.25
 Aroma_JAR              41.25  21.25     37.5
 Color_JAR              41.25  18.75       40
 Orange_Flavor_JAR      46.25   17.5    36.25
 Sweetness_JAR             40     25       35
 Mouthfeel_JAR          43.75  13.75     42.5
 Aftertaste_JAR         41.25     20    38.75,
                   Not Enough    JAR Too Much
 Flavor_JAR              42.5     15     42.5
 Aroma_JAR               37.5  18.75    43.75
 Color_JAR               37.5     15     47.5
 Orange_Flavor_JAR       47.5   12.5       40
 Sweetness_JAR          41.25  58.75        0
 Mouthfeel_JAR          21.25  26.25    38.75
 Aftertaste_JAR         38.75   17.5    43.75)
```

In [3]:

```
#Finding Correlation for directional JAR Scores to Overall Liking

# Transform JAR scores into weighted dummy variables for "Not Enough" and "Too M
for attribute in jar_attributes:
    rawdata[f'{attribute}_Not_Enough'] = rawdata[attribute].apply(lambda x: 1 if
    rawdata[f'{attribute}_Too_Much'] = rawdata[attribute].apply(lambda x: 1 if x

# Create the formula for the regression model
dummy_variables = [f'{attr}_Not_Enough' for attr in jar_attributes] + [f'{attr}_
formula = 'Overall_Liking ~ ' + ' + '.join(dummy_variables)

# Function to fit the model for a specific product
def fit_model_for_product(product_data):
    model = ols(formula, data=product_data).fit()
    print(f"Model summary for {product_data['Product'].iloc[0]}:")
    print(model.summary())
    print("\n")
    return model

# Separate the data for each product
original_data = rawdata[rawdata['Product'] == 'Original']
new_formulation_data = rawdata[rawdata['Product'] == 'New Formulation']

# Fit the models for each product
original_model = fit_model_for_product(original_data)
new_formulation_model = fit_model_for_product(new_formulation_data)

# Save the model summaries to text files
with open('Original_Product_Multiple_Regression_Summary.txt', 'w') as f:
    f.write(original_model.summary().as_text())

with open('New_Formulation_Product_Multiple_Regression_Summary.txt', 'w') as f:
    f.write(new_formulation_model.summary().as_text())
```

```
Model summary for Original:
                            OLS Regression Results
==============================================================================
Dep. Variable:         Overall_Liking   R-squared:                       0.143
Model:                            OLS   Adj. R-squared:                 -0.042
Method:                 Least Squares   F-statistic:                    0.7739
Date:                Fri, 31 May 2024   Prob (F-statistic):              0.693
Time:                        23:01:31   Log-Likelihood:                -179.22
No. Observations:                  80   AIC:                             388.4
Df Residuals:                      65   BIC:                             424.2
Df Model:                          14
```

```
Covariance Type:            nonrobust
================================================================================
================
                                   coef    std err          t      P>|t|      
[0.025      0.975]
--------------------------------------------------------------------------------
----------------
Intercept                        4.1625      1.623      2.565      0.013      
0.921       7.404
Flavor_JAR_Not_Enough            0.2190      1.086      0.202      0.841      -
1.950       2.388
Aroma_JAR_Not_Enough             0.9054      0.851      1.064      0.291      -
0.795       2.605
Color_JAR_Not_Enough            -1.1877      0.995     -1.194      0.237      -
3.174       0.799
Orange_Flavor_JAR_Not_Enough    -0.5474      0.917     -0.597      0.553      -
2.380       1.285
Sweetness_JAR_Not_Enough         0.9587      0.916      1.046      0.299      -
0.871       2.788
Mouthfeel_JAR_Not_Enough         0.1133      1.010      0.112      0.911      -
1.903       2.130
Aftertaste_JAR_Not_Enough        1.0976      0.929      1.181      0.242      -
0.758       2.953
Flavor_JAR_Too_Much              0.7081      0.904      0.783      0.436      -
1.098       2.514
Aroma_JAR_Too_Much               0.1422      0.934      0.152      0.880      -
1.723       2.008
Color_JAR_Too_Much              -0.8374      0.890     -0.941      0.350      -
2.614       0.940
Orange_Flavor_JAR_Too_Much       0.0656      0.962      0.068      0.946      -
1.855       1.986
Sweetness_JAR_Too_Much           1.5840      0.910      1.741      0.086      -
0.233       3.401
Mouthfeel_JAR_Too_Much           0.2151      0.980      0.220      0.827      -
1.741       2.171
Aftertaste_JAR_Too_Much          0.2335      1.041      0.224      0.823      -
1.846       2.313
==============================================================================
Omnibus:                        4.809   Durbin-Watson:                   2.082
Prob(Omnibus):                  0.090   Jarque-Bera (JB):                2.886
Skew:                          -0.261   Prob(JB):                        0.236
Kurtosis:                       2.229   Cond. No.                         11.8
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.


Model summary for New Formulation:
                            OLS Regression Results                            
==============================================================================
Dep. Variable:         Overall_Liking   R-squared:                       0.428
Model:                            OLS   Adj. R-squared:                  0.315
Method:                 Least Squares   F-statistic:                     3.799
Date:                Fri, 31 May 2024   Prob (F-statistic):           0.000152
Time:                        23:01:31   Log-Likelihood:                -171.66
No. Observations:                  80   AIC:                             371.3
Df Residuals:                      66   BIC:                             404.7
Df Model:                          13                                         
Covariance Type:            nonrobust                                         
================================================================================
================
                                   coef    std err          t      P>|t|      
[0.025      0.975]
```

```
--------------------------------------------------------------------------------
----------------
Intercept                        7.0844      1.388      5.106      0.000       
4.314       9.855
Flavor_JAR_Not_Enough           -0.0867      0.847     -0.102      0.919      -
1.779       1.605
Aroma_JAR_Not_Enough             0.3469      0.870      0.399      0.691      -
1.390       2.084
Color_JAR_Not_Enough             0.6205      0.833      0.745      0.459      -
1.042       2.283
Orange_Flavor_JAR_Not_Enough    -0.8550      0.867     -0.986      0.328      -
2.587       0.877
Sweetness_JAR_Not_Enough        -4.9018      0.820     -5.977      0.000      -
6.539      -3.264
Mouthfeel_JAR_Not_Enough         0.1296      1.455      0.089      0.929      -
2.775       3.034
Aftertaste_JAR_Not_Enough       -0.0426      0.819     -0.052      0.959      -
1.677       1.592
Flavor_JAR_Too_Much             -0.0380      0.832     -0.046      0.964      -
1.699       1.623
Aroma_JAR_Too_Much               0.8392      0.876      0.958      0.342      -
0.910       2.589
Color_JAR_Too_Much              -0.6538      0.953     -0.686      0.495      -
2.557       1.249
Orange_Flavor_JAR_Too_Much      -0.6641      0.862     -0.770      0.444      -
2.385       1.057
Sweetness_JAR_Too_Much        3.963e-15   3.82e-15      1.037      0.304   -3.67
e-15    1.16e-14
Mouthfeel_JAR_Too_Much          -0.0712      0.768     -0.093      0.926      -
1.604       1.462
Aftertaste_JAR_Too_Much         -0.7135      0.907     -0.786      0.434      -
2.525       1.098
==============================================================================
Omnibus:                        5.303   Durbin-Watson:                   1.922
Prob(Omnibus):                  0.071   Jarque-Bera (JB):                2.424
Skew:                          -0.068   Prob(JB):                        0.298
Kurtosis:                       2.158   Cond. No.                     5.27e+17
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The smallest eigenvalue is 6.37e-34. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

In [4]:

```
#Intensity score analysis
intensity_attributes = [col for col in rawdata.columns if 'Intensity' in col]
intensity_means = rawdata.groupby('Product')[intensity_attributes].mean().round(

# Separating the data for each product
original_data = rawdata[rawdata['Product'] == 'Original']
new_formulation_data = rawdata[rawdata['Product'] == 'New Formulation']

# Performing paired t-test and Wilcoxon signed-rank test for each intensity attr
t_test_results = {}
wilcoxon_test_results = {}

for attribute in intensity_attributes:
    t_stat, t_p_value = ttest_rel(original_data[attribute], new_formulation_data
    t_test_results[attribute] = (t_stat, t_p_value)
    w_stat, w_p_value = wilcoxon(original_data[attribute], new_formulation_data[
    wilcoxon_test_results[attribute] = (w_stat, w_p_value)

# Creating dataframes for the results
t_test_results_df = pd.DataFrame.from_dict(t_test_results, orient='index', colum
wilcoxon_test_results_df = pd.DataFrame.from_dict(wilcoxon_test_results, orient=

intensity_means, t_test_results_df, wilcoxon_test_results_df
#
```

Out[4]:

```
(                 Sweetness_Intensity  Orange_Flavor_Intensity  \
 Product
 New Formulation                 4.94                     4.95
 Original                        4.99                     5.24

                  Aftertaste_Intensity
 Product
 New Formulation                  5.22
 Original                         5.75  ,
                           t_stat   p_value
 Sweetness_Intensity      0.177445  0.859613
 Orange_Flavor_Intensity  0.811947  0.419262
 Aftertaste_Intensity     1.588946  0.116067,
                          w_stat   p_value
 Sweetness_Intensity      1062.0  0.779423
 Orange_Flavor_Intensity  1221.0  0.366285
 Aftertaste_Intensity      731.5  0.122381)
```

In [ ]:

In [ ]: