Suvo_Ganguli_Assignment_5.1

February 7, 2024

1 Assignment 5.1

Name: Subhabrata (Suvo) Ganguli

Date: Feb 7, 2024

For this assignment, you will refer to the textbook to solve the practice exercises. **Use Python to answer any coding problems (not R, even if indicated in your textbook).** Use Jupyter Notebook, Google Colab, or a similar software program to complete your assignment. Submit your answers as a **PDF or HTML** file. As a best practice, always label your axes and provide titles for any graphs generated on this assignment. Round all quantitative answers to 2 decimal places.

1.1 Problem 5.1.

Introducing notation for a parameter, state the following hypotheses in terms of the parameter values and indicate whether it is a null hypothesis or an alternative hypothesis.

(a) The proportion of all adults in the UK who favor legalized gambling equals 0.50.

Proportion of all adults in the UK favoring legalized gambling as "pi"

Null hypothesis (H0): pi = 0.50

Alternate hypothesis (HA): pi != 0.50

(b) The correlation for Australian adults between smoking (number of cigarettes per day) and blood pressure is positive.

Let the correlation between smoking (number of cigarettes per day) and blood pressure for Australian adults = c

Null hypothesis (H0): c = 0

Alternate hypothesis (HA): c > 0

(c) The mean grade point average this year of all college graduates in the U.S. is the same for females and males.

Let the the mean grade point average this year of female college graduates in the U.S = mu_f

Let the the mean grade point average this year of male college graduates in the U.S = mu_m

Null hyothesis (H0): mu_f = mu_m

Alternate hypothesis (HA): mu_f != mu_m

1.2 Problem 5.6.

Before a Presidential election, polls are taken in two swing states. The Republican candidate was preferred by 59 of the 100 people sampled in state A and by 525 of 1000 sampled in state B. Treat these as independent binomial samples, where the parameter π is the population proportion voting Republican in the state.

(a) If we can treat these polls as if the samples were random, use significance tests of H_0: $\pi = 0.50$ against $H_a : \pi > 0.50$ to determine which state has greater evidence supporting a Republican victory. Explain your reasoning.

Let us use the following notations where - H0 is the null hypothesis - HA is the alternate hypothesis

1.2.1 State A:

H0: pi_A = 0.50
HA: pi_A > 0.50

1.2.2 State B:

H0: pi_B = 0.50
HA: pi_B > 0.50

1.2.3 For State A:

pA = 59/100 = 0.59

Standard error = SE = 0.50/sqrt(100) = 0.05

t-statistic = (0.59 - 0.50)/0.05 = 1.80

Therefore p-value = 0.0359.

Since 0.0359 < 0.05, we reject the null hypothesis and conclude a Republican victory in State A.

1.2.4 For State B:

pA = 525/1000 = 0.525

Standard error = SE = 0.50/sqrt(1000) = 0.016

t-statistic = (0.525 - 0.50)/0.016 = 1.56

Therefore p-value = 0.0569.

Since 0.0569 > 0.05, we cannot reject the null hypothesis and we cannot conclude a Republican victory in State B.

[42]:
```
# Sample sizes and number
n_A = 100
n_B = 1000
x_A = 59
x_B = 525

# Proportions
```

```
p_A = x_A / n_A
p_B = x_B / n_B

# Null hypothesis
p = 0.50

# standard errors
se_A = np.sqrt(p * (1 - p) / n_A)
se_B = np.sqrt(p * (1 - p) / n_B)

print("se_A = %.3f" % se_A)
print("se_B = %.3f" % se_B)

# z-values
z_A = (p_A - p) / se_A
z_B = (p_B - p) / se_B

print("z_A = %.3f" % z_A)
print("z_B = %.3f" % z_B)

# p-values
p_value_A = 1 - norm.cdf(z_A)
p_value_B = 1 - norm.cdf(z_B)

# print p-values
print("p-value for State A = ", p_value_A)
print("p-value for State B = ", p_value_B)
```

```
se_A = 0.050
se_B = 0.016
z_A = 1.800
z_B = 1.581
p-value for State A =  0.03593031911292588
p-value for State B =  0.05692314900332884
```

(b) Conduct a Bayesian analysis to answer the question in (a) by finding in each case the posterior $P(\pi < 0.50)$, corresponding to the P- value in (a). Use beta(50, 50) priors, which have standard deviation 0.05 and reflect the pollster's strong prior belief that π almost surely is between 0.35 and 0.65. Explain any differences between conclusions.

The difference between (a) and (b) is because in (b) the Bayesian approach calculates the posterior probability from the observed data, while in (a), only the null hypothesis is used

[43]:
```
import numpy as np
import scipy.stats as stats

# Prior and likelihood parameters
alpha_prior = 50
```

```python
beta_prior = 50
xA = 59
xB = 525
nA = 100
nB = 1000

# Prior and Posterior beta
prior_A = stats.beta(alpha_prior, beta_prior)
prior_B = stats.beta(alpha_prior, beta_prior)
posterior_A = stats.beta(alpha_prior + xA, beta_prior - xA + nA)
posterior_B = stats.beta(alpha_prior + xB, beta_prior - xB + nB)

# Prior probability
prob_A_prior = prior_A.cdf(0.50)
prob_B_prior = prior_B.cdf(0.50)

# Posterior probability
prob_A_posterior = posterior_A.cdf(0.50)
prob_B_posterior = posterior_B.cdf(0.50)

print("Prior probability (State A) = ", prob_A_prior)
print("Prior probability (State B) = ", prob_B_prior)
print("Posterior probability (State A) = ", prob_A_posterior)
print("Posterior probability (State B) = ", prob_B_posterior)
```

```
Prior probability (State A) =  0.5000000000000004
Prior probability (State B) =  0.5000000000000004
Posterior probability (State A) =  0.10092130164939611
Posterior probability (State B) =  0.06572759751525574
```

1.3 Problem 5.8.

For the `Students` data file at the text website, analyze political ideology.

(a) Test whether the population mean μ differs from 4.0, the moderate response. Report the P-value, and interpret. Make a conclusion using α - level = 0.05.

The p-value = 2.48e-05

[44]:
```python
# import libraries
import pandas as pd

# read data
df = pd.read_csv("Students.csv")

# see data head
df.head()
```

[44]:

```
   subject  gender  age  hsgpa  cogpa  dhome  dres    tv  sport  news  aids  \
0        1       0   32    2.2    3.5      0   5.0   3.0      5     0     0
1        2       1   23    2.1    3.5   1200   0.3  15.0      7     5     6
2        3       1   27    3.3    3.0   1300   1.5   0.0      4     3     0
3        4       1   35    3.5    3.2   1500   8.0   5.0      5     6     3
4        5       0   23    3.1    3.5   1600  10.0   6.0      6     3     0

   veg  affil  ideol  relig  abor  affirm  life
0    0      2      6      2     0       0     1
1    1      1      2      1     1       1     3
2    1      1      2      2     1       1     3
3    0      3      4      1     1       1     2
4    0      3      1      0     1       0     2
```

[45]:

```python
# import library

from scipy.stats import ttest_1samp

# calculate t-statistic and p-value for the data
mu_test = 4
t_statistic, p_value = ttest_1samp(df["ideol"], mu_test)

# print
print("t-statistic:", t_statistic)
print("p-value:", p_value)
```

```
t-statistic: -4.576584373210669
p-value: 2.4837197305408817e-05
```

(b) Construct the 95% confidence interval for μ. Explain how results relate to those of the test in (a).

The confidence interval does not contain the mu value of 4. This is consistent with the fact that we rejected the null hypothesis.

[46]:

```python
# calculate the confidence interval
mean = np.mean(df["ideol"])
std = np.std(df["ideol"])
se = std/len(df["ideol"])

ci_low = mean - 1.96*se
ci_hi = mean + 1.96*se

# print confidence intervals
print("Confidence Interval Low = %.2f" % ci_low)
print("Confidence Interval Hi = %.2f" % ci_hi)
```

```
Confidence Interval Low = 2.98
Confidence Interval Hi = 3.09
```

1.4 Problem 5.10.

A study of sheep mentioned in Exercise 1.27 analyzed whether the sheep survived for a year from the original observation time (1 = yes, 0 = no) as a function of their weight (kg) at the original observation. Stating any assumptions including the conceptual population of interest, use a t test with the data in the Sheep data file at the text website to compare mean weights of the sheep that survived and did not survive. Interpret the P-value.

Assumptions: - The two groups (survived and did not survive) are independent. - The weights of sheep in each group are approximately normally distributed. - The variances of the two groups are roughly equal.

Hypothesis: - Null: mean_survived = mean_not_survived - Alternate: mean_survided != mean_not_survived

Since the p-value is < 0.05, we reject the null hypothesis. That is, there is significant difference between the weights of the sheep survived vs not survived.

[47]:
```python
# import libraries
import pandas as pd

# read data
df = pd.read_csv("Sheep.csv", delimiter=r"\s+")

# see data head
df.head()
```

[47]:

	sheep	weight	survival
0	1	20.8	0
1	2	23.0	1
2	3	28.0	1
3	4	27.5	1
4	5	26.0	0

[48]:
```python
# Groups
survived = df[df["survival"] == 1]["weight"]
not_survived = df[df["survival"] == 0]["weight"]

print("Mean survived = ", np.mean(survived))
print("Mean not_survived = ", np.mean(not_survived))

# t-test
t, p = stats.ttest_ind(survived, not_survived, equal_var=True)

# Print results
print("t-statistic = ", t)
print("p-value = ", p)
```

```
Mean survived =  20.645917387127763
Mean not_survived =  15.998113207547169
```

```
t-statistic =  14.500255633782931
p-value =  2.1010522406615758e-44
```

1.5 Problem 5.11.

Use descriptive statistics and significance tests to compare the population mean political ideology for each pair of groups in Table 5.2 using the `Polid` data file. Summarize results using P-values and using a non-technical explanation.

See code in the cells below.

Since the p-values are < 0.05, we can interpret that there are significant differences in the means between each group.

[49]:
```python
# import libraries
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import ttest_ind
from itertools import combinations

# read data
df = pd.read_csv("Polid.dat", delimiter=r"\s+")

# see data head
print(df.head())

# Convert 'Category' column to categorical
df['races'] = pd.Categorical(df['race'])

# Print the number of unique categories
print("Number of unique categories:", df['races'].unique())
```

```
       race  ideology
1  hispanic         1
2  hispanic         1
3  hispanic         1
4  hispanic         1
5  hispanic         1
Number of unique categories: ['hispanic', 'black', 'white']
Categories (3, object): ['black', 'hispanic', 'white']
```

[50]:
```python
# group the data for plotting
df_race_ideology = df.groupby(['race', 'ideology']).size().unstack(fill_value=0)

# Plot the data
df_race_ideology.plot(kind='bar', stacked=True)
plt.title('Race vs Ideology')
plt.xlabel('Race')
plt.ylabel('Count')
```

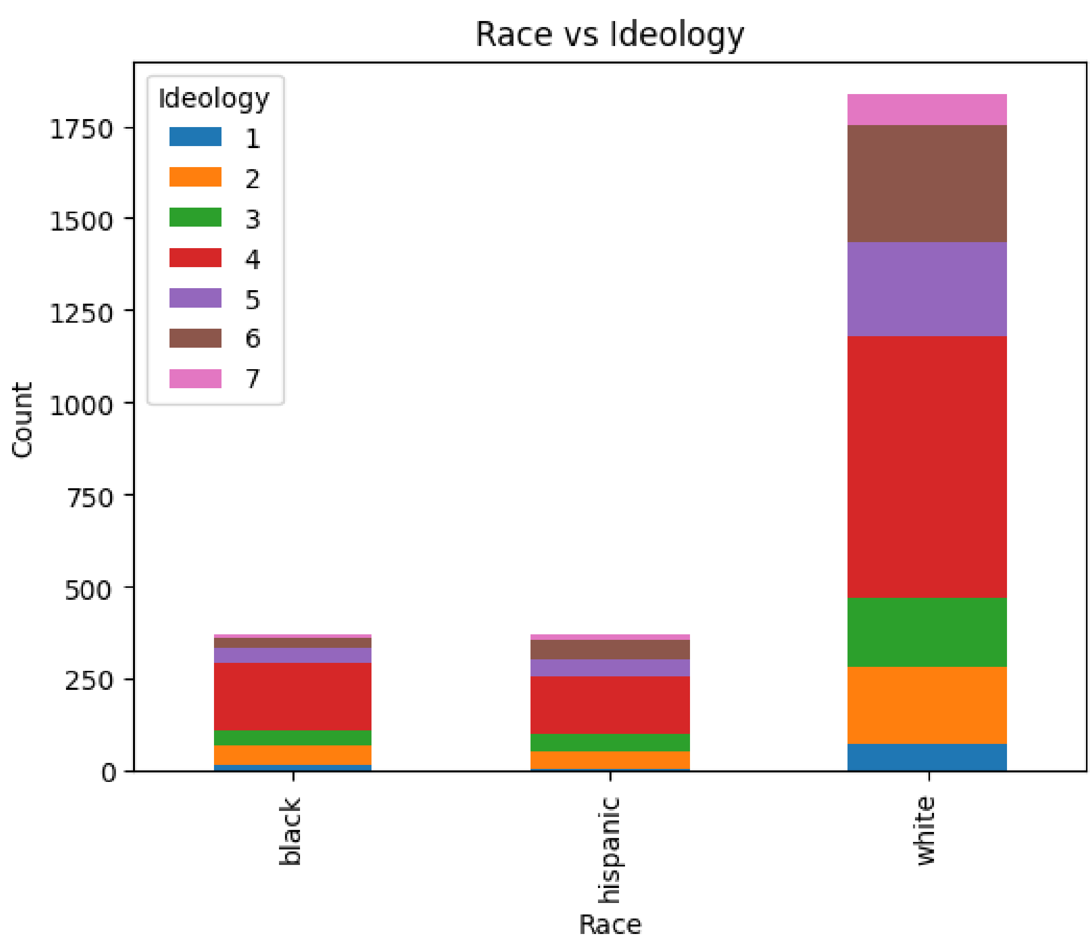
```
plt.legend(title='Ideology')
plt.show()

# describe the statistics
stats = df_race_ideology.describe()
print(stats)

# group the data by 'race' and 'ideology'
df_grouped = df.groupby(['race', 'ideology'])

# Perform significance tests between pairs of groups
groups = df_grouped.groups.keys()
group_pairs = combinations(groups, 2)

for group1, group2 in group_pairs:
    ideology_1 = df_grouped.get_group(group1)['ideology']
    ideology_2 = df_grouped.get_group(group2)['ideology']
    t_statistic, p_value = ttest_ind(ideology_1, ideology_2)
    print(f"For {group1} and {group2}, t-statistic: {t_statistic}, p-value:␣
 ↪{p_value}")
```



```
ideology          1           2           3           4           5  \
count      3.000000    3.000000    3.000000    3.000000    3.000000
mean      31.333333  103.333333   92.666667  347.333333  117.666667
std       36.501142   91.522311   84.316863  310.042470  123.313962
min        5.000000   49.000000   42.000000  155.000000   43.000000
25%       10.500000   50.500000   44.000000  168.500000   46.500000
50%       16.000000   52.000000   46.000000  182.000000   50.000000
75%       44.500000  130.500000  118.000000  443.500000  155.000000
max       73.000000  209.000000  190.000000  705.000000  260.000000

ideology           6          7
count       3.000000   3.000000
mean      129.666667  36.333333
std       160.125992  41.307788
min        25.000000  11.000000
25%        37.500000  12.500000
50%        50.000000  14.000000
75%       182.000000  49.000000
max       314.000000  84.000000
For ('black', 1) and ('black', 2), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('black', 3), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('black', 4), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('black', 5), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('black', 6), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('black', 7), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('hispanic', 1), t-statistic: nan, p-value: nan
For ('black', 1) and ('hispanic', 2), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('hispanic', 3), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('hispanic', 4), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('hispanic', 5), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('hispanic', 6), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('hispanic', 7), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('white', 1), t-statistic: nan, p-value: nan
For ('black', 1) and ('white', 2), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('white', 3), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('white', 4), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('white', 5), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('black', 1) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('black', 3), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('black', 4), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('black', 5), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('black', 6), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('black', 7), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('hispanic', 1), t-statistic: inf, p-value: 0.0
```

```
For ('black', 2) and ('hispanic', 2), t-statistic: nan, p-value: nan
For ('black', 2) and ('hispanic', 3), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('hispanic', 4), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('hispanic', 5), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('hispanic', 6), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('hispanic', 7), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('white', 1), t-statistic: inf, p-value: 0.0
For ('black', 2) and ('white', 2), t-statistic: nan, p-value: nan
For ('black', 2) and ('white', 3), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('white', 4), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('white', 5), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('black', 2) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('black', 3) and ('black', 4), t-statistic: -inf, p-value: 0.0
For ('black', 3) and ('black', 5), t-statistic: -inf, p-value: 0.0
For ('black', 3) and ('black', 6), t-statistic: -inf, p-value: 0.0
For ('black', 3) and ('black', 7), t-statistic: -inf, p-value: 0.0
For ('black', 3) and ('hispanic', 1), t-statistic: inf, p-value: 0.0
For ('black', 3) and ('hispanic', 2), t-statistic: inf, p-value: 0.0
For ('black', 3) and ('hispanic', 3), t-statistic: nan, p-value: nan
For ('black', 3) and ('hispanic', 4), t-statistic: -inf, p-value: 0.0
For ('black', 3) and ('hispanic', 5), t-statistic: -inf, p-value: 0.0
For ('black', 3) and ('hispanic', 6), t-statistic: -inf, p-value: 0.0
For ('black', 3) and ('hispanic', 7), t-statistic: -inf, p-value: 0.0
For ('black', 3) and ('white', 1), t-statistic: inf, p-value: 0.0
For ('black', 3) and ('white', 2), t-statistic: inf, p-value: 0.0
For ('black', 3) and ('white', 3), t-statistic: nan, p-value: nan
For ('black', 3) and ('white', 4), t-statistic: -inf, p-value: 0.0
For ('black', 3) and ('white', 5), t-statistic: -inf, p-value: 0.0
For ('black', 3) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('black', 3) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('black', 4) and ('black', 5), t-statistic: -inf, p-value: 0.0
For ('black', 4) and ('black', 6), t-statistic: -inf, p-value: 0.0
For ('black', 4) and ('black', 7), t-statistic: -inf, p-value: 0.0
For ('black', 4) and ('hispanic', 1), t-statistic: inf, p-value: 0.0
For ('black', 4) and ('hispanic', 2), t-statistic: inf, p-value: 0.0
For ('black', 4) and ('hispanic', 3), t-statistic: inf, p-value: 0.0
For ('black', 4) and ('hispanic', 4), t-statistic: nan, p-value: nan
For ('black', 4) and ('hispanic', 5), t-statistic: -inf, p-value: 0.0
For ('black', 4) and ('hispanic', 6), t-statistic: -inf, p-value: 0.0
For ('black', 4) and ('hispanic', 7), t-statistic: -inf, p-value: 0.0
For ('black', 4) and ('white', 1), t-statistic: inf, p-value: 0.0
For ('black', 4) and ('white', 2), t-statistic: inf, p-value: 0.0
For ('black', 4) and ('white', 3), t-statistic: inf, p-value: 0.0
For ('black', 4) and ('white', 4), t-statistic: nan, p-value: nan
For ('black', 4) and ('white', 5), t-statistic: -inf, p-value: 0.0
For ('black', 4) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('black', 4) and ('white', 7), t-statistic: -inf, p-value: 0.0
```

```
For ('black', 5) and ('black', 6), t-statistic: -inf, p-value: 0.0
For ('black', 5) and ('black', 7), t-statistic: -inf, p-value: 0.0
For ('black', 5) and ('hispanic', 1), t-statistic: inf, p-value: 0.0
For ('black', 5) and ('hispanic', 2), t-statistic: inf, p-value: 0.0
For ('black', 5) and ('hispanic', 3), t-statistic: inf, p-value: 0.0
For ('black', 5) and ('hispanic', 4), t-statistic: inf, p-value: 0.0
For ('black', 5) and ('hispanic', 5), t-statistic: nan, p-value: nan
For ('black', 5) and ('hispanic', 6), t-statistic: -inf, p-value: 0.0
For ('black', 5) and ('hispanic', 7), t-statistic: -inf, p-value: 0.0
For ('black', 5) and ('white', 1), t-statistic: inf, p-value: 0.0
For ('black', 5) and ('white', 2), t-statistic: inf, p-value: 0.0
For ('black', 5) and ('white', 3), t-statistic: inf, p-value: 0.0
For ('black', 5) and ('white', 4), t-statistic: inf, p-value: 0.0
For ('black', 5) and ('white', 5), t-statistic: nan, p-value: nan
For ('black', 5) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('black', 5) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('black', 6) and ('black', 7), t-statistic: -inf, p-value: 0.0
For ('black', 6) and ('hispanic', 1), t-statistic: inf, p-value: 0.0
For ('black', 6) and ('hispanic', 2), t-statistic: inf, p-value: 0.0
For ('black', 6) and ('hispanic', 3), t-statistic: inf, p-value: 0.0
For ('black', 6) and ('hispanic', 4), t-statistic: inf, p-value: 0.0
For ('black', 6) and ('hispanic', 5), t-statistic: inf, p-value: 0.0
For ('black', 6) and ('hispanic', 6), t-statistic: nan, p-value: nan
For ('black', 6) and ('hispanic', 7), t-statistic: -inf, p-value: 0.0
For ('black', 6) and ('white', 1), t-statistic: inf, p-value: 0.0
For ('black', 6) and ('white', 2), t-statistic: inf, p-value: 0.0
For ('black', 6) and ('white', 3), t-statistic: inf, p-value: 0.0
For ('black', 6) and ('white', 4), t-statistic: inf, p-value: 0.0
For ('black', 6) and ('white', 5), t-statistic: inf, p-value: 0.0
For ('black', 6) and ('white', 6), t-statistic: nan, p-value: nan
For ('black', 6) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('black', 7) and ('hispanic', 1), t-statistic: inf, p-value: 0.0
For ('black', 7) and ('hispanic', 2), t-statistic: inf, p-value: 0.0
For ('black', 7) and ('hispanic', 3), t-statistic: inf, p-value: 0.0
For ('black', 7) and ('hispanic', 4), t-statistic: inf, p-value: 0.0
For ('black', 7) and ('hispanic', 5), t-statistic: inf, p-value: 0.0
For ('black', 7) and ('hispanic', 6), t-statistic: inf, p-value: 0.0
For ('black', 7) and ('hispanic', 7), t-statistic: nan, p-value: nan
For ('black', 7) and ('white', 1), t-statistic: inf, p-value: 0.0
For ('black', 7) and ('white', 2), t-statistic: inf, p-value: 0.0
For ('black', 7) and ('white', 3), t-statistic: inf, p-value: 0.0
For ('black', 7) and ('white', 4), t-statistic: inf, p-value: 0.0
For ('black', 7) and ('white', 5), t-statistic: inf, p-value: 0.0
For ('black', 7) and ('white', 6), t-statistic: inf, p-value: 0.0
For ('black', 7) and ('white', 7), t-statistic: nan, p-value: nan
For ('hispanic', 1) and ('hispanic', 2), t-statistic: -inf, p-value: 0.0
For ('hispanic', 1) and ('hispanic', 3), t-statistic: -inf, p-value: 0.0
For ('hispanic', 1) and ('hispanic', 4), t-statistic: -inf, p-value: 0.0
```

```
For ('hispanic', 1) and ('hispanic', 5), t-statistic: -inf, p-value: 0.0
For ('hispanic', 1) and ('hispanic', 6), t-statistic: -inf, p-value: 0.0
For ('hispanic', 1) and ('hispanic', 7), t-statistic: -inf, p-value: 0.0
For ('hispanic', 1) and ('white', 1), t-statistic: nan, p-value: nan
For ('hispanic', 1) and ('white', 2), t-statistic: -inf, p-value: 0.0
For ('hispanic', 1) and ('white', 3), t-statistic: -inf, p-value: 0.0
For ('hispanic', 1) and ('white', 4), t-statistic: -inf, p-value: 0.0
For ('hispanic', 1) and ('white', 5), t-statistic: -inf, p-value: 0.0
For ('hispanic', 1) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('hispanic', 1) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('hispanic', 2) and ('hispanic', 3), t-statistic: -inf, p-value: 0.0
For ('hispanic', 2) and ('hispanic', 4), t-statistic: -inf, p-value: 0.0
For ('hispanic', 2) and ('hispanic', 5), t-statistic: -inf, p-value: 0.0
For ('hispanic', 2) and ('hispanic', 6), t-statistic: -inf, p-value: 0.0
For ('hispanic', 2) and ('hispanic', 7), t-statistic: -inf, p-value: 0.0
```

```
/usr/local/lib/python3.11/site-packages/scipy/stats/_axis_nan_policy.py:523:
RuntimeWarning: Precision loss occurred in moment calculation due to
catastrophic cancellation. This occurs when the data are nearly identical.
Results may be unreliable.
  res = hypotest_fun_out(*samples, **kwds)
```

```
For ('hispanic', 2) and ('white', 1), t-statistic: inf, p-value: 0.0
For ('hispanic', 2) and ('white', 2), t-statistic: nan, p-value: nan
For ('hispanic', 2) and ('white', 3), t-statistic: -inf, p-value: 0.0
For ('hispanic', 2) and ('white', 4), t-statistic: -inf, p-value: 0.0
For ('hispanic', 2) and ('white', 5), t-statistic: -inf, p-value: 0.0
For ('hispanic', 2) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('hispanic', 2) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('hispanic', 3) and ('hispanic', 4), t-statistic: -inf, p-value: 0.0
For ('hispanic', 3) and ('hispanic', 5), t-statistic: -inf, p-value: 0.0
For ('hispanic', 3) and ('hispanic', 6), t-statistic: -inf, p-value: 0.0
For ('hispanic', 3) and ('hispanic', 7), t-statistic: -inf, p-value: 0.0
For ('hispanic', 3) and ('white', 1), t-statistic: inf, p-value: 0.0
For ('hispanic', 3) and ('white', 2), t-statistic: inf, p-value: 0.0
For ('hispanic', 3) and ('white', 3), t-statistic: nan, p-value: nan
For ('hispanic', 3) and ('white', 4), t-statistic: -inf, p-value: 0.0
For ('hispanic', 3) and ('white', 5), t-statistic: -inf, p-value: 0.0
For ('hispanic', 3) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('hispanic', 3) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('hispanic', 4) and ('hispanic', 5), t-statistic: -inf, p-value: 0.0
For ('hispanic', 4) and ('hispanic', 6), t-statistic: -inf, p-value: 0.0
For ('hispanic', 4) and ('hispanic', 7), t-statistic: -inf, p-value: 0.0
For ('hispanic', 4) and ('white', 1), t-statistic: inf, p-value: 0.0
For ('hispanic', 4) and ('white', 2), t-statistic: inf, p-value: 0.0
For ('hispanic', 4) and ('white', 3), t-statistic: inf, p-value: 0.0
For ('hispanic', 4) and ('white', 4), t-statistic: nan, p-value: nan
For ('hispanic', 4) and ('white', 5), t-statistic: -inf, p-value: 0.0
For ('hispanic', 4) and ('white', 6), t-statistic: -inf, p-value: 0.0
```

```
For ('hispanic', 4) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('hispanic', 5) and ('hispanic', 6), t-statistic: -inf, p-value: 0.0
For ('hispanic', 5) and ('hispanic', 7), t-statistic: -inf, p-value: 0.0
For ('hispanic', 5) and ('white', 1), t-statistic: inf, p-value: 0.0
For ('hispanic', 5) and ('white', 2), t-statistic: inf, p-value: 0.0
For ('hispanic', 5) and ('white', 3), t-statistic: inf, p-value: 0.0
For ('hispanic', 5) and ('white', 4), t-statistic: inf, p-value: 0.0
For ('hispanic', 5) and ('white', 5), t-statistic: nan, p-value: nan
For ('hispanic', 5) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('hispanic', 5) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('hispanic', 6) and ('hispanic', 7), t-statistic: -inf, p-value: 0.0
For ('hispanic', 6) and ('white', 1), t-statistic: inf, p-value: 0.0
For ('hispanic', 6) and ('white', 2), t-statistic: inf, p-value: 0.0
For ('hispanic', 6) and ('white', 3), t-statistic: inf, p-value: 0.0
For ('hispanic', 6) and ('white', 4), t-statistic: inf, p-value: 0.0
For ('hispanic', 6) and ('white', 5), t-statistic: inf, p-value: 0.0
For ('hispanic', 6) and ('white', 6), t-statistic: nan, p-value: nan
For ('hispanic', 6) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('hispanic', 7) and ('white', 1), t-statistic: inf, p-value: 0.0
For ('hispanic', 7) and ('white', 2), t-statistic: inf, p-value: 0.0
For ('hispanic', 7) and ('white', 3), t-statistic: inf, p-value: 0.0
For ('hispanic', 7) and ('white', 4), t-statistic: inf, p-value: 0.0
For ('hispanic', 7) and ('white', 5), t-statistic: inf, p-value: 0.0
For ('hispanic', 7) and ('white', 6), t-statistic: inf, p-value: 0.0
For ('hispanic', 7) and ('white', 7), t-statistic: nan, p-value: nan
For ('white', 1) and ('white', 2), t-statistic: -inf, p-value: 0.0
For ('white', 1) and ('white', 3), t-statistic: -inf, p-value: 0.0
For ('white', 1) and ('white', 4), t-statistic: -inf, p-value: 0.0
For ('white', 1) and ('white', 5), t-statistic: -inf, p-value: 0.0
For ('white', 1) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('white', 1) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('white', 2) and ('white', 3), t-statistic: -inf, p-value: 0.0
For ('white', 2) and ('white', 4), t-statistic: -inf, p-value: 0.0
For ('white', 2) and ('white', 5), t-statistic: -inf, p-value: 0.0
For ('white', 2) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('white', 2) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('white', 3) and ('white', 4), t-statistic: -inf, p-value: 0.0
For ('white', 3) and ('white', 5), t-statistic: -inf, p-value: 0.0
For ('white', 3) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('white', 3) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('white', 4) and ('white', 5), t-statistic: -inf, p-value: 0.0
For ('white', 4) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('white', 4) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('white', 5) and ('white', 6), t-statistic: -inf, p-value: 0.0
For ('white', 5) and ('white', 7), t-statistic: -inf, p-value: 0.0
For ('white', 6) and ('white', 7), t-statistic: -inf, p-value: 0.0
```

1.6 Problem 5.14 (a).

The `Income` data file at the book's website shows annual incomes in thousands of dollars for subjects in three racial-ethnic groups in the U.S.

(a) Stating all assumptions including the relative importance of each, show all steps of a significance test for comparing population mean incomes of Blacks and Hispanics. Interpret.

Assumptions: - The sample collected should be using random sampling method - The individuals from each group should be independent of each other - The incomes for each group should approximately follow a normal distribution - The variances of the incomes for each group should be approximately equal

For significance test, we use the t-statistics and p-value. Since the p-value is not less than 0.05, we cannot reject the null hypothesis tha the incomes of the two races are equal.

[51]:
```
# read data
df = pd.read_csv("Income.dat", delimiter=r"\s+")

# see data head
print(df.head())

# Convert 'Category' column to categorical
df['races'] = pd.Categorical(df['race'])

# Print the number of unique categories
print("Number of unique categories:", df['races'].unique())
```

```
   income  education race
0      16         10    B
1      18          7    B
2      26          9    B
3      16         11    B
4      34         14    B
Number of unique categories: ['B', 'H', 'W']
Categories (3, object): ['B', 'H', 'W']
```

[52]:
```
# Extract incomes for the races
income_black = df[df['race'] == 'B']['income']
income_hispanic = df[df['race'] == 'H']['income']

# calculate the mean and standard deviations
mean_black = np.mean(income_black)
std_black = np.std(income_black)

mean_hispanic = np.mean(income_hispanic)
std_hispanic = np.std(income_hispanic)

print("Mean income for Black = %.2f" % mean_black)
print("Std income for Black = %.2f" % std_black)
```

```
print("Mean income for Hispanic = %.2f" % mean_hispanic)
print("Std income for Hispanic = %.2f" % std_hispanic)

# Calculate t-statistics
t_statistic, p_value = ttest_ind(income_black, income_hispanic)

print("t-statistic = %.5f" % t_statistic)
print("p-value = %.5f" % p_value)

# Significance level
alpha = 0.05

if p_value < alpha:
    print("Reject null hypothesis that the mean income are equal")
else:
    print("Fail to reject null hypothesis that the mean income are equal")
```

```
Mean income for Black = 27.75
Std income for Black = 12.86
Mean income for Hispanic = 31.00
Std income for Hispanic = 12.35
t-statistic = -0.67962
p-value = 0.50233
Fail to reject null hypothesis that the mean income are equal
```

1.7 Problem 5.15.

A recent report [39] estimated mean adult heights in the U.S. of 175.4 *cm* (69.1 inches) for men and 161.7 *cm* (63.7 inches) for women, with standard deviation about 7 *cm* for each group. For all finishers in the Boston Marathon since 1972, the time to finish has a mean of 221 minutes for men and 248 minutes for women, each with a standard deviation of about 40 minutes. According to the effect size, is the difference between men and women greater for height or for marathon times? Explain.

Since the ratio of mean difference between height and corresponding standard deviation compared to that for marathon time, the difference between men and women is greater for height.

[58]:
```
# statistics for height
mean_diff_height = 175.4 - 161.7
std_height = 7
ratio_height = mean_diff_height / std_height
print("Ratio for height = %.2f" % ratio_height)

# statistics for marathon time
mean_diff_time = 248 - 221
std_time = 40
ratio_time = mean_diff_time / std_time
print("Ratio for time = %.2f" % ratio_time)
```

```
Ratio for height = 1.96
Ratio for time = 0.68
```

1.8 Problem 5.17.

Ideally, results of a statistical analysis should not depend greatly on a single observation. In a sensitivity study, we re-do the analysis after deleting an outlier from the data set or changing its value to a more typical value and checking whether results change much. For the anorexia data analysis in Section 5.3.2, the weight change of 20.9 pounds for the cb group was a severe outlier. Suppose this observation was actually 2.9 pounds but recorded incorrectly. Find the P-value for testing $H_0 : \mu 1 = \mu 2$ against $H_a : \mu 1 \neq \mu 2$ with and without that observation. Summarize its influence.

Since I could not find any of the rows where the weight difference is 20.9, I decided two add a new row containing the outlier. (Note: the maximum value of the difference in weight is 9.1). The weight before was taken as the mean of the before-weight, abnd 20.9 was subtracted to get the after-weight. Similarly, a row was added to a new copy of the original dataframe where the weight difference is 2.9. All analysis is done with the two datasets - df_outlier and df_no_outlier.

The null hypothesis is H0: the mean difference in weight are equal. The alternate hypothesis is HA: the mean differnce are not equal.

With the above, I performed the t-statistic test and calculates the p-value.

I observe that the p-value is 0.7684. Since the p-value is > 0.05, we cannot reject the hypothesis. That is, the mean with and without the outlier are significantly different.

[59]:
```
# read data
df = pd.read_csv("Anorexia.dat", delimiter=r"\s+")

# get the cd data
df = df[df["therapy"] == 'cb']

# calculate the weight difference
df["diff"] = df["before"] - df["after"]

# see data head
print(df.head())

# try finding if there is an observation with a weight difference of 20.9
value = 20.9
result = df[df['diff'] == value]
print(result)

# find the maximum weight difference
print(np.max(df['diff']))

# find mean weight before
mean_before = np.mean(df["before"])
```

```python
# find after-weights for the two cases - with and without outlier
data_outlier = mean_before - 20.9
data_no_outlier = mean_before - 2.9

# add the two new observations since there is no observation where the␣
↪difference is 20.9

# (a) with outlier
print(df.tail())
df_tmp = pd.DataFrame([[71,'cb',mean_before,data_outlier,20.9]],␣
↪columns=['subject','therapy','before','after','diff'])
df_outlier = pd.concat([df,df_tmp])
print(df_outlier.tail())

# (b) without outlier
df_tmp = pd.DataFrame([[71,'cb',mean_before,data_no_outlier,2.9]],␣
↪columns=['subject','therapy','before','after','diff'])
df_no_outlier = pd.concat([df,df_tmp])
print(df_no_outlier.tail())

# hypothesis test comparing with and without outlier
t_statistic, p_value = ttest_ind(df_outlier["diff"], df_no_outlier["diff"])

print("p-value = %.4f" % p_value)

# significance level
alpha = 0.05

# hypothesis
if p_value < alpha:
    print("Reject null hypothesis")
else:
    print("Cannot reject null hypothesis")
```

```
   subject therapy  before  after  diff
0        1      cb    80.5   82.2  -1.7
1        2      cb    84.9   85.6  -0.7
2        3      cb    81.5   81.4   0.1
3        4      cb    82.6   81.9   0.7
4        5      cb    79.9   76.4   3.5
Empty DataFrame
Columns: [subject, therapy, before, after, diff]
Index: []
9.100000000000009
    subject therapy  before  after  diff
24       25      cb    83.3   85.2  -1.9
25       26      cb    79.7   83.6  -3.9
```

```
26        27       cb     84.5    84.6  -0.1
27        28       cb     80.8    96.2 -15.4
28        29       cb     87.4    86.7   0.7
    subject therapy      before       after  diff
25        26       cb  79.700000  83.600000  -3.9
26        27       cb  84.500000  84.600000  -0.1
27        28       cb  80.800000  96.200000 -15.4
28        29       cb  87.400000  86.700000   0.7
0         71       cb  82.689655  61.789655  20.9
    subject therapy      before       after  diff
25        26       cb  79.700000  83.600000  -3.9
26        27       cb  84.500000  84.600000  -0.1
27        28       cb  80.800000  96.200000 -15.4
28        29       cb  87.400000  86.700000   0.7
0         71       cb  82.689655  79.789655   2.9
p-value = 0.7684
Cannot reject null hypothesis
```

1.9 Problem 5.19.

In the 2018 General Social Survey, when asked whether they believed in life after death, 1017 of 1178 females said *yes*, and 703 of 945 males said *yes*. Test that the population proportions are equal for females and males. Report and interpret the P-value.

Null hypothesis: The population proportions are equal for females and males. Alternate hypothesis: The population proportions are not equal for females and males.

We calculate the two sample proportion z-test with the female and male populations.

The z-score = 6.973

and the p-value ~= 0.

Since, the p-value is less than 0.05, we reject the null hypothesis that the proportions are equal for females and males.

[60]:
```python
# import library
from scipy.stats import norm

# female data
female_yes = 1017
female_total = 1178

# male data
male_yes = 703
male_total = 945

# proportions
pf = female_yes/female_total
pm = male_yes/male_total
```

```
print("pf = %.3f" % pf)
print("pm = %.3f" % pm)

p = (female_yes + male_yes) / (female_total + male_total)
print("p = %.3f" % p)

# two sample proportion z-test
z = (pf - pm)/np.sqrt(p * (1 - p) * (1/female_total + 1/male_total))
print("z = %.3f" % z)

p_value = 2 * (1 - norm.cdf(abs(z)))
print("p-value = %.3f" % p_value)

# alpha for t-statistic test
alpha = 0.05

# hypothesis
if p_value < alpha:
    print("Reject null hypothesis")
else:
    print("Cannot reject null hypothesis")
```

```
pf = 0.863
pm = 0.744
p = 0.810
z = 6.973
p-value = 0.000
Reject null hypothesis
```

1.10 Problem 5.23.

Use the Happy data file from the 2018 General Social Survey at the text website to form a contingency table that cross classifies happiness with gender. For $ H_0 $: independence between happiness and gender:

(a) Conduct and interpret the chi-squared test.

The code and results for the contingency table and the chi-squared test is given below.

The chi-squared value = 0.9165, and p-value = 0.6323

Since the p-value is > 0.05, we cannot reject the null hypothesis and we conclude that happiness and gender are have no association.

[61]:
```
# import libraries
import pandas as pd
from scipy.stats import chi2_contingency

# load the Happy data file
df = pd.read_csv("Happy.dat", delimiter = r"\s+")
```

```
# create a contingency table
contingency_table = pd.crosstab(df['happiness'], df['gender'])
print("Contigency table:")
print(contingency_table)

# Do the chi-squared test
chi2_stat, p_value, dof, expected = chi2_contingency(contingency_table)
print("Chi-squared statistic:", chi2_stat)
print("P-value:", p_value)

# significant level
alpha = 0.05

# interpretation
if p_value < alpha:
    print("Reject the null hypothesis.")
else:
    print("Cannot reject the null hypothesis.")
```

```
Contigency table:
gender     female  male
happiness
1             353   295
2             642   553
3             153   146
Chi-squared statistic: 0.9165315892565513
P-value: 0.6323793708278013
Cannot reject the null hypothesis.
```

(b) Show the estimated expected frequencies and standardized residuals, and form a mosaic plot. Explain how they are consistent with the result of the chi-squared test.

The expected frequencies and standardized residuals are calculated in the code below.

Note that the all the tiles in the mosaic plot are of similar sizes in terms of male vs female (rows). However, from the happiness perspective (columns), the mid-level (level = 2) is of larger size than the other two levels. This corresponds with the expected frequency table where the happiness level 2 for females and males are 642 and 553 - which are much larger than the other two levels. Also, the frequencies of the happiness level 3 is much smaller than level 1 and 2.

In other words, there is some dependency of the data with the happiness level. This is why we cannot reject the null hypothesis.

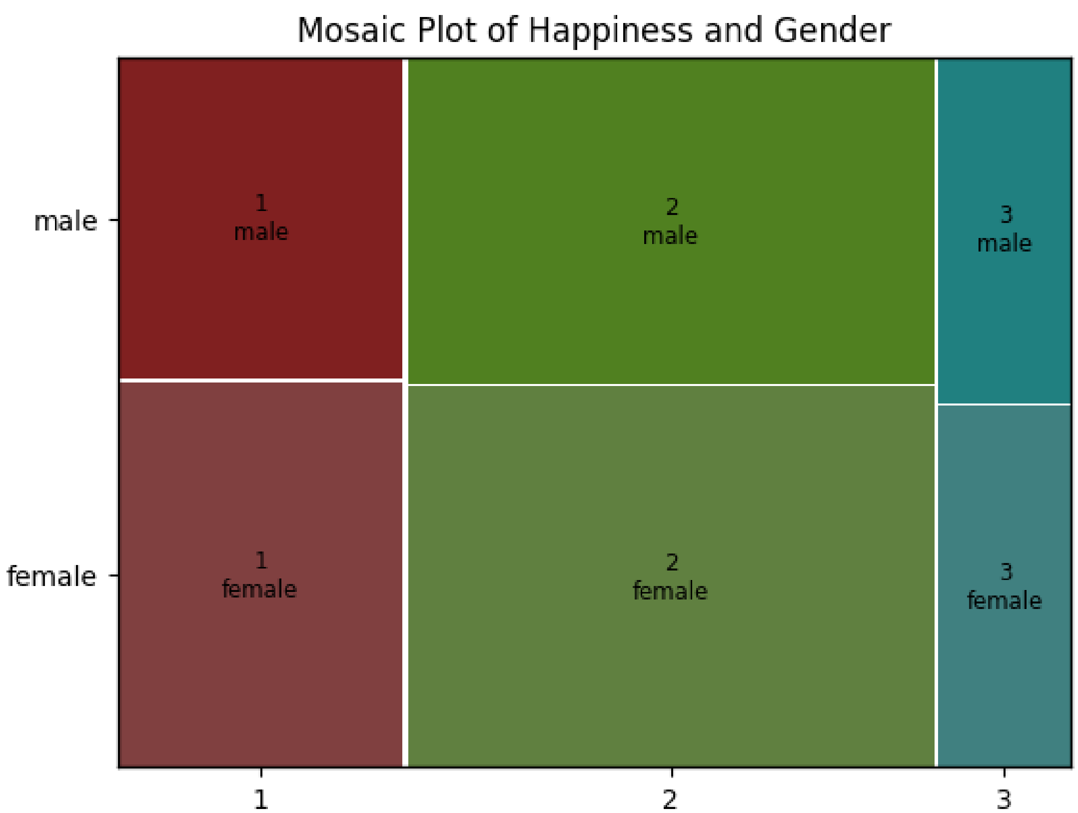
[62]:
```
# import library
import statsmodels.graphics.mosaicplot as sp

# expected frequency
print("Expected frequencies = ", expected)
```

```python
# standardized residuals
residuals = (contingency_table - expected) / np.sqrt(expected)
print("Standardized residuals:\n", residuals)

# mosaic plot
sp.mosaic(contingency_table.stack(), title="Mosaic Plot of Happiness and␣
↪Gender")
plt.show()
```

```
Expected frequencies =  [[347.29411765 300.70588235]
 [640.45751634 554.54248366]
 [160.24836601 138.75163399]]
Standardized residuals:
 gender       female      male
happiness
1           0.306178 -0.329042
2           0.060950 -0.065502
3          -0.572589  0.615348
```



[39] See www.cdc.gov/nchs/data/nhsr/nhsr122-508.pdf and https://doi.org/10.1371/journal.pone.