5. After EBRT tumor size and before_brachy_categor

April 29, 2022

[1]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import statsmodels.api as sm
%matplotlib inline
pd.set_option('display.max_rows', None)
import scipy.stats as stats
from scipy.stats import shapiro
import copy
import statsmodels.stats.api as sms
from scipy.stats import wilcoxon
sns.set(style="darkgrid")
import test_mod as mod
from scipy.stats import chi2_contingency
```

1 Descriptive analysis of attributes

1.1 Importing data

[2]:
```python
df = pd.read_csv(r"C:\Users\gdbt0\OneDrive\GitHub\Projects\cancer␣
 ↪research\data\processed\research_variables.csv")
df.head()
```

[2]:
```
    id tumour_stage diagnos_tumor_size after EBRT_tumor_size  \
0  101           2B                 4
1  102           2B                 4                     2
2  103           3B                >4                    >2
3  104           2B                >4                    >2
4  105           3B                >4                     2

   before_brachy_categor  post_treatment_response
0                      1                        0
1                      1                        0
2                      2                        2
3                      2                        0
4                      2                        1
```

[3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 120 entries, 0 to 119
Data columns (total 6 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   id                        120 non-null    int64
 1   tumour_stage              120 non-null    object
 2   diagnos_tumor_size        120 non-null    object
 3   after EBRT_tumor_size     120 non-null    object
 4   before_brachy_categor     120 non-null    int64
 5   post_treatment_response   120 non-null    int64
dtypes: int64(3), object(3)
memory usage: 5.8+ KB
```

1.2 After EBRT tumor size and before_brachy_categor

[4]:
```
#Transforming data
primary_var_name = 'after EBRT_tumor_size'
secondary_var_name = 'before_brachy_categor'

primary_vs_secondary = df[[primary_var_name,secondary_var_name]].copy()
primary_vs_secondary.dropna(inplace=True)
primary_vs_secondary[primary_var_name] = primary_vs_secondary[primary_var_name].
 ↪astype('category')
```

1.2.1 after EBRT_tumor_size

[5]:
```
# Declaration of the variable
primary_var = df[primary_var_name].copy()
primary_var.dropna(inplace=True)
primary_var_value_counts = primary_var.value_counts()
```

[6]:
```
## Finding the frequency distribution of categories in the variable
print("The frequency distribution of categories  in {}.".
 ↪format(primary_var_name))
print("In total, there are {} categories.".format(primary_var.value_counts().
 ↪count()))
primary_var.value_counts()
```

```
The frequency distribution of categories  in after EBRT_tumor_size.
In total, there are 3 categories.
```

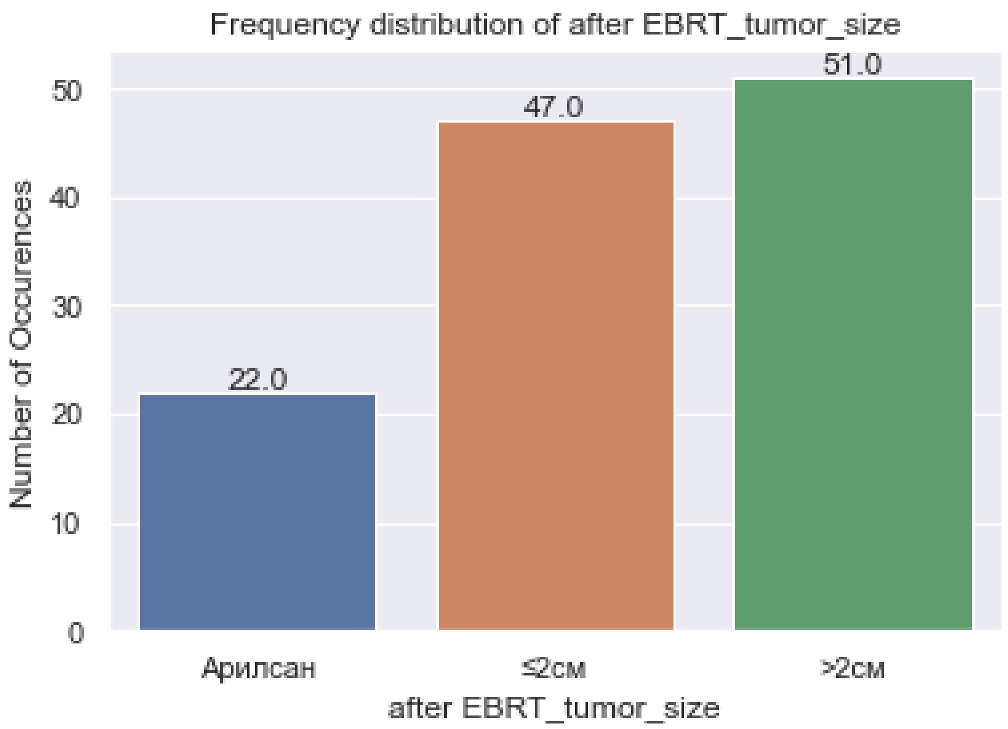
[6]:
```
>2        51
 2       47
        22
```

```
Name: after EBRT_tumor_size, dtype: int64
```

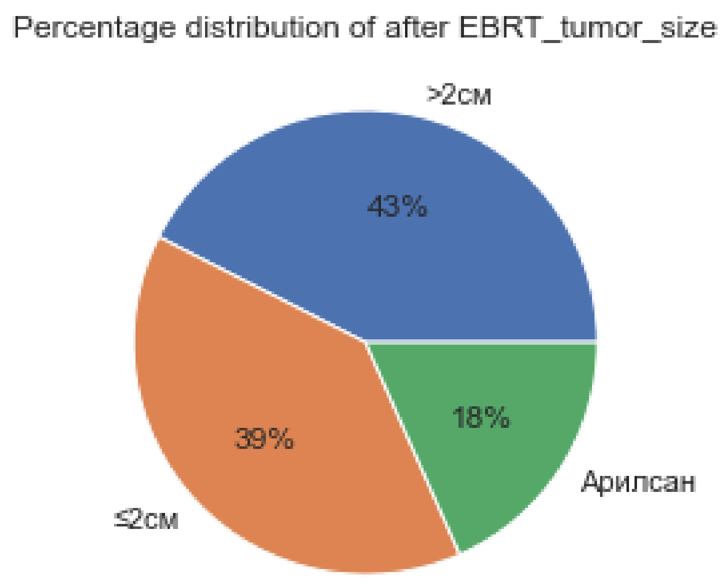
Frequency distribution of the categorical features to a barplot.

[7]:
```python
# Barplot
plot = sns.countplot(primary_var)
mod.show_values(plot)
plt.title("Frequency distribution of {}".format(primary_var_name))
plt.ylabel('Number of Occurences', fontsize=12)
plt.xlabel('{}'.format(primary_var_name),fontsize=12)
plt.show()
```

```
C:\Users\gdbt0\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



[8]:
```python
#Making a pie chart
plt.title("Percentage distribution of {}".format(primary_var_name))
plt.pie(primary_var_value_counts, labels=primary_var_value_counts.index,␣
↪autopct='%.0f%%')
plt.show()
```



1.2.2 before_brachy_categor

```
[9]: # Declaration of the variable
     secondary_var = df[secondary_var_name].copy()
     secondary_var.dropna(inplace=True)
     secondary_var_value_counts = secondary_var.value_counts()
```

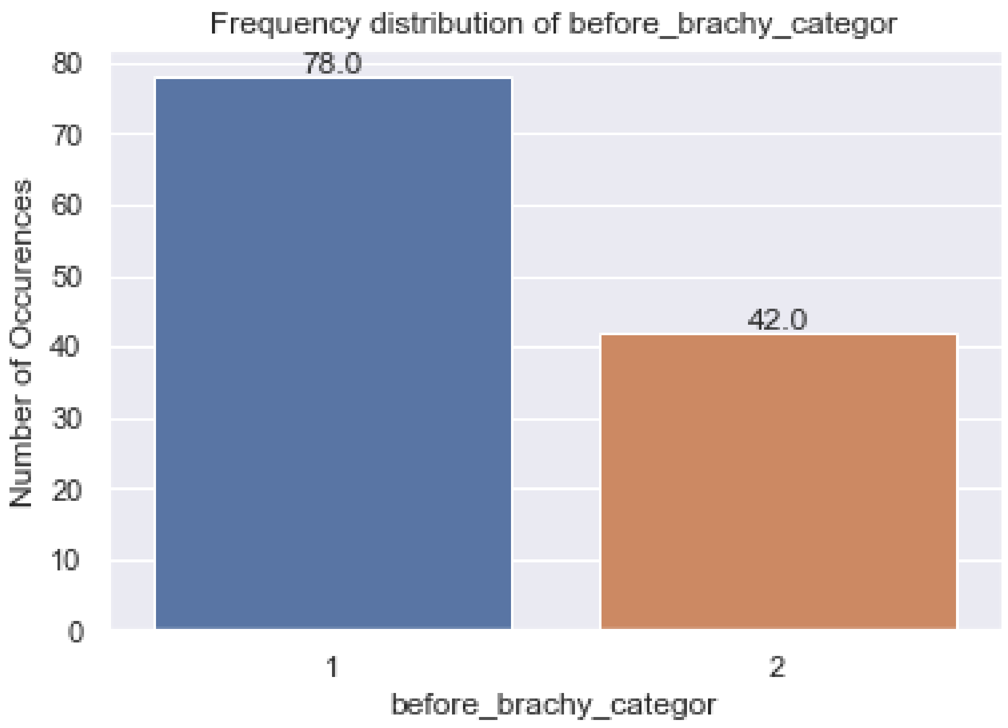
```
[10]: ## Finding the frequency distribution of categories in the variable
      print("The frequency distribution of categories  in {}.".
       ↪format(secondary_var_name))
      print("In total, there are {} categories.".format(secondary_var.value_counts().
       ↪count()))
      secondary_var.value_counts()
```

```
The frequency distribution of categories  in before_brachy_categor.
In total, there are 2 categories.
```

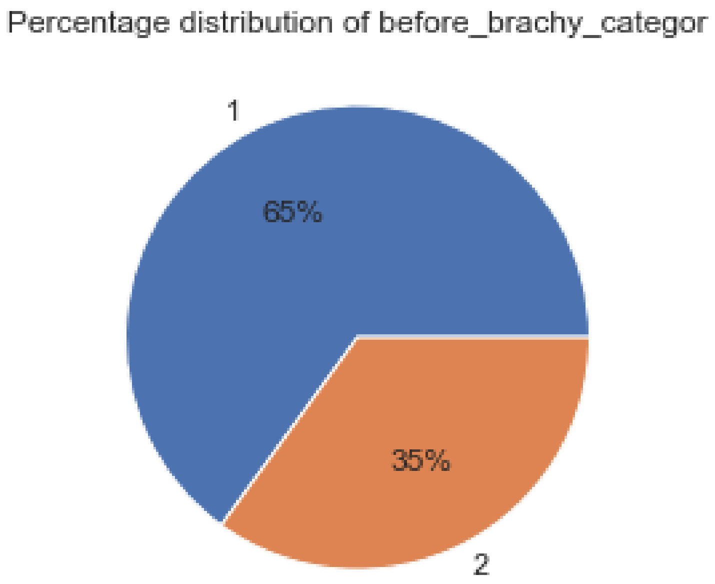
```
[10]: 1    78
      2    42
      Name: before_brachy_categor, dtype: int64
```

```
[11]: # Barplot
      plot = sns.countplot(secondary_var)
      mod.show_values(plot)
      plt.title("Frequency distribution of {}".format(secondary_var_name))
      plt.ylabel('Number of Occurences', fontsize=12)
      plt.xlabel('{}'.format(secondary_var_name),fontsize=12)
      plt.show()
```

```
C:\Users\gdbt0\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



[12]:
```
#Pie chart
plt.title("Percentage distribution of {}".format(secondary_var_name))
plt.pie(secondary_var_value_counts, labels=secondary_var_value_counts.index,␣
 ↪autopct='%.0f%%')
plt.show()
```



1.2.3 Analysis of groups

Creates groups by after EBRT_tumor_size

[13]:
```
firstGroup = primary_vs_secondary[primary_vs_secondary[primary_var_name] ==␣
↪" 2 "][secondary_var_name]
secondGroup = primary_vs_secondary[primary_vs_secondary[primary_var_name] ==␣
↪">2 "][secondary_var_name]
```

1.2.4 belowTwo

The group where the After EBRT tumor size is below 2cm

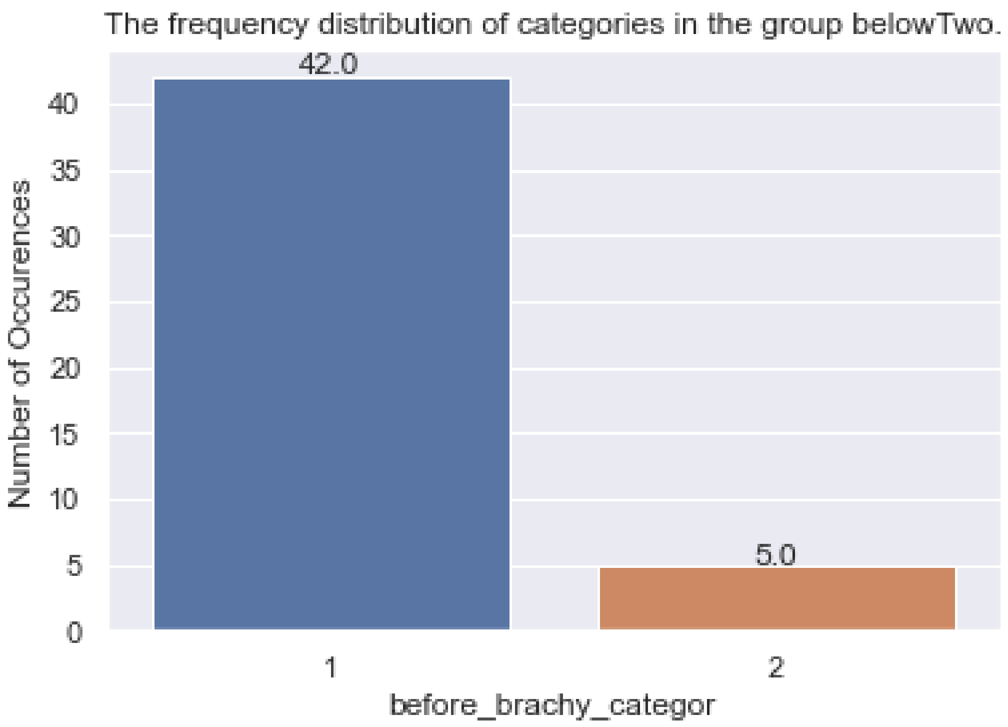
[14]:
```
## Finding the frequency distribution of categories in the variable
print("The frequency distribution of categories in the group belowTwo.")
print("In total, there are {} categories.".format(firstGroup.value_counts().
↪count()))
firstGroup.value_counts()
```

```
The frequency distribution of categories in the group belowTwo.
In total, there are 2 categories.
```

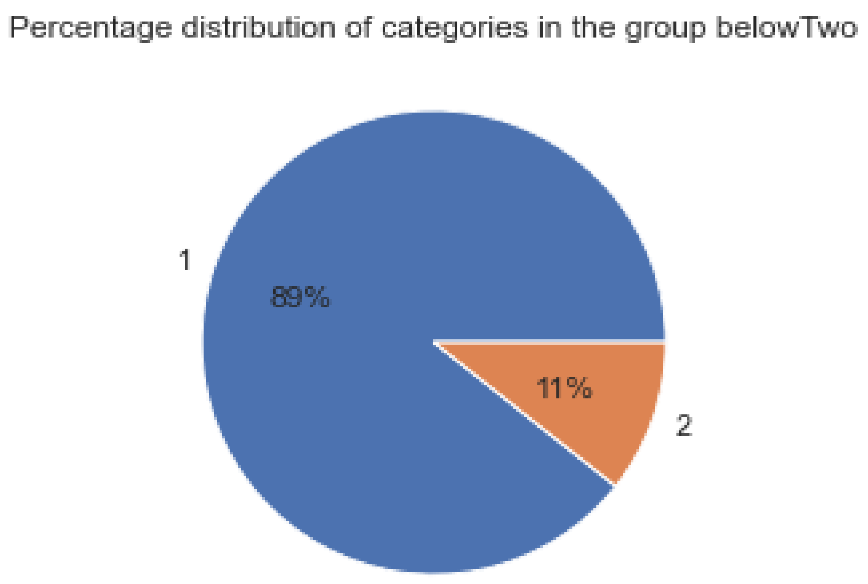
[14]:
```
1    42
2     5
Name: before_brachy_categor, dtype: int64
```

[15]:
```python
# Barplot
plot = sns.countplot(firstGroup)
mod.show_values(plot)
plt.title("The frequency distribution of categories in the group belowTwo.")
plt.ylabel('Number of Occurences', fontsize=12)
plt.xlabel('{}'.format(secondary_var_name),fontsize=12)
plt.show()
```

```
C:\Users\gdbt0\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



[16]:
```python
#Pie chart
plt.title("Percentage distribution of categories in the group belowTwo")
plt.pie(firstGroup.value_counts(), labels=firstGroup.value_counts().index,␣
 ↪autopct='%.0f%%')
plt.show()
```



1.2.5 aboveTwo

The group where the After EBRT tumor size is above 2cm

[17]:
```
## Finding the frequency distribution of categories in the variable
print("The frequency distribution of categories in the group aboveTwo.")
print("In total, there are {} categories.".format(secondGroup.value_counts().
 ↪count()))
secondGroup.value_counts()
```

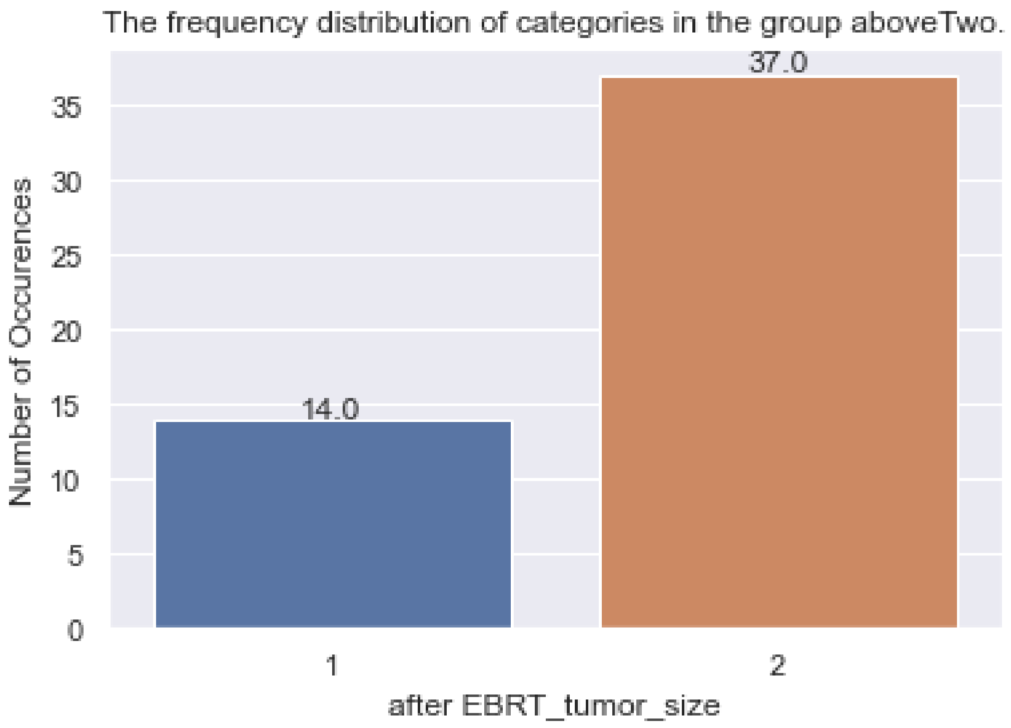
```
The frequency distribution of categories in the group aboveTwo.
In total, there are 2 categories.
```

[17]:
```
2    37
1    14
Name: before_brachy_categor, dtype: int64
```

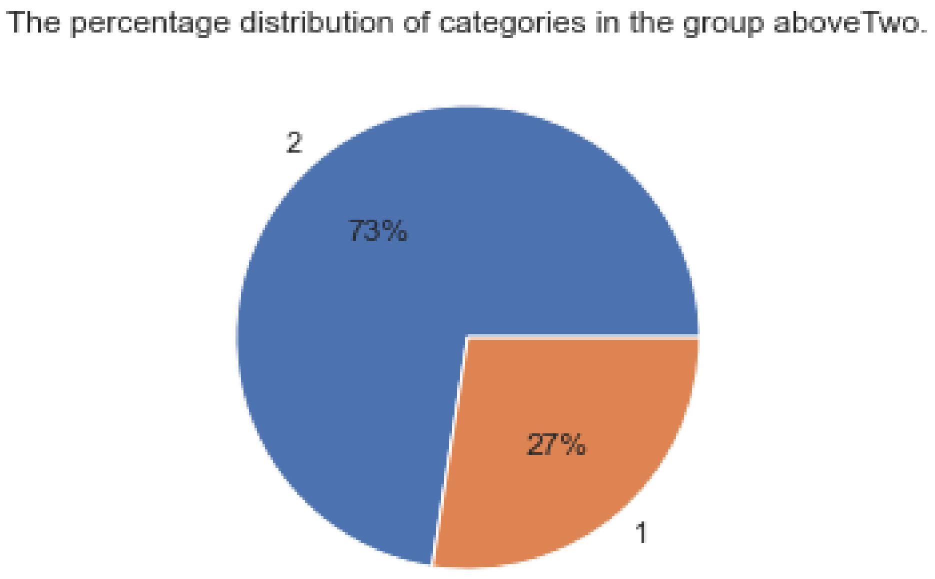
[18]:
```
# Barplot
plot = sns.countplot(secondGroup)
mod.show_values(plot)
plt.title("The frequency distribution of categories in the group aboveTwo.")
plt.ylabel('Number of Occurences', fontsize=12)
plt.xlabel('{}'.format(primary_var_name),fontsize=12)
plt.show()
```

```
C:\Users\gdbt0\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
```

```
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



```
[19]: #Pie chart
      plt.title("The percentage distribution of categories in the group aboveTwo.")
      plt.pie(secondGroup.value_counts(), labels=secondGroup.value_counts().index,␣
       ↪autopct='%.0f%%')
      plt.show()
```



1.2.6 Chi square test for Independence

Test that the categorical values of the two groups are different to each other.

Reference

[1] Chi-Bar-Squared. Retrieved March 9, 2022 from:

[2] Johns Hopkins.

[3] Kenney, J. F. and Keeping, E. S. Mathematics of Statistics, Pt. 2, 2nd ed. Princeton, NJ: Van Nostrand, 1951.

[4] https://www.statisticshowto.com/probability-and-statistics/chi-square/

[5] https://www.askpython.com/python/examples/chi-square-test

1.2.7 Uses the pandas crosstab() function to create a contingency table of the two selected variables

[20]:
```
chisqt = pd.crosstab(primary_var, secondary_var, margins=True)
print(chisqt)
```

```
before_brachy_categor   1   2  All
after EBRT_tumor_size
>2                      14  37   51
                      22   0   22
 2                     42   5   47
All                       78  42  120
```

```
[21]: value = np.array([chisqt.iloc[0][0:5].values, chisqt.iloc[1][0:5].values])
      print(value)
```

```
[[14 37 51]
 [22  0 22]]
```

```
[22]: chi_stat = chi2_contingency(value)[0]
      p_value = chi2_contingency(value)[1]
      degrees_of_freedom = chi2_contingency(value)[2]
      significance_level = 0.05
```

The null hypothesis: The grouping variables have no association or correlation amongst them. #### The alternate hypothesis The variables are associated with each other and happen to have a correlation between the variables.

```
[23]: print("The p-value of the test is: {}".format(p_value) + "\nThe test statistic␣
       ↪is: {}".format(chi_stat) + "\nThe degrees of freedom is: {}".
       ↪format(degrees_of_freedom)
         )
```

```
The p-value of the test is: 9.37661543720416e-08
The test statistic is: 32.36492374727669
The degrees of freedom is: 2
```

```
[24]: if p_value <= significance_level:
          print('Reject NULL HYPOTHESIS')
      else:
          print('Accept NULL HYPOTHESIS')
```

```
Reject NULL HYPOTHESIS
```