DATA INSPECTION/CLEANING

```
data = pd.read_csv('/Users/tonydao/Documents/PythonProjects/LungCancerProject/lungCancer.csv')
print(data.head())
print(data.info())
  GENDER  AGE  SMOKING  YELLOW_FINGERS  ANXIETY  PEER_PRESSURE  \
0      M   69        1               2        2              1
1      M   74        2               1        1              1
2      F   59        1               1        1              2
3      M   63        2               2        2              1
4      F   63        1               2        1              1

   CHRONIC DISEASE  FATIGUE   ALLERGY   WHEEZING  ALCOHOL CONSUMING  COUGHING  \
0                1        2         1          2                  2         2
1                2        2         2          1                  1         1
2                1        2         1          2                  1         2
3                1        1         1          1                  2         1
4                1        1         1          2                  1         2

   SHORTNESS OF BREATH  SWALLOWING DIFFICULTY  CHEST PAIN LUNG_CANCER
0                    2                      2           2         YES
1                    2                      2           2         YES
2                    2                      1           2          NO
3                    1                      2           2          NO
4                    2                      1           1          NO
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 309 entries, 0 to 308
Data columns (total 16 columns):
Data columns (total 16 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   GENDER                 309 non-null    object
 1   AGE                    309 non-null    int64
 2   SMOKING                309 non-null    int64
 3   YELLOW_FINGERS         309 non-null    int64
 4   ANXIETY                309 non-null    int64
 5   PEER_PRESSURE          309 non-null    int64
 6   CHRONIC DISEASE        309 non-null    int64
 7   FATIGUE                309 non-null    int64
 8   ALLERGY                309 non-null    int64
 9   WHEEZING               309 non-null    int64
 10  ALCOHOL CONSUMING      309 non-null    int64
 11  COUGHING               309 non-null    int64
 12  SHORTNESS OF BREATH    309 non-null    int64
 13  SWALLOWING DIFFICULTY  309 non-null    int64
 14  CHEST PAIN             309 non-null    int64
 15  LUNG_CANCER            309 non-null    object
dtypes: int64(14), object(2)
memory usage: 38.8+ KB
None
```

- Here we can see what cols we are working with and the data type that exist within each cols.
- We have 16 cols.
  - 2 are type of Objects
  - 14 are floats.
- I would change any funky label names with something simple.

- **Checking for missing values and duplicates**

```
print(data.isnull().sum())
print(data.duplicated().sum())
GENDER                   0
AGE                      0
SMOKING                  0
YELLOW_FINGERS           0
ANXIETY                  0
PEER_PRESSURE            0
CHRONIC DISEASE          0
FATIGUE                  0
ALLERGY                  0
WHEEZING                 0
ALCOHOL CONSUMING        0
COUGHING                 0
SHORTNESS OF BREATH      0
SWALLOWING DIFFICULTY    0
CHEST PAIN               0
LUNG_CANCER              0
dtype: int64
33
```

- Here we see non of our cols contain any nulls.
- We do have 33 duplicates.

```python
duplicates = data[data.duplicated()]
print(duplicates)
```

```
    GENDER  AGE  SMOKING  YELLOW_FINGERS  ANXIETY  PEER_PRESSURE  \
99       M   56        2               1        1              1
100      M   58        2               1        1              1
117      F   51        2               2        2              2
199      F   55        2               1        1              2
212      M   58        2               1        1              1
223      M   63        2               2        2              1
256      M   60        2               1        1              1
275      M   64        2               2        2              2
284      M   58        2               2        2              2
285      F   58        2               2        2              2
286      F   63        1               1        1              1
287      F   51        2               2        2              2
288      F   61        1               2        2              2
289      F   61        2               1        1              1
290      M   76        2               1        1              1
291      M   71        2               2        2              1
292      M   69        1               1        2              1
293      F   56        2               2        2              1
294      M   67        1               1        1              2
295      F   54        2               2        2              1
296      M   63        1               2        1              1
297      F   47        2               2        1              2
298      M   62        2               1        2              1
299      M   65        2               2        2              2
300      F   63        2               2        2              2
301      M   64        1               2        2              2
302      F   65        2               2        2              2
303      M   51        1               2        1              1
304      F   56        1               1        1              2
305      M   70        2               1        1              1
306      M   58        2               1        1              1
307      M   67        2               1        2              1
308      M   62        1               1        1              2
```

- If we ever need the duplicates ones only then this variable holds it and we can display the specific duplicated rows.

```python
data = data.drop_duplicates()
```

```python
print(data.duplicated().sum())
```

```
0
```

- We drop any duplicated rows, and we can confirm our results.

- **Standardize Categorical Data**
  - Our 'LUNG_CANCER' col is 'YES/NO'. Since all of our others rows are integer. We can represent this cols as '1/0' to allow data analysis be conducted more efficiently.

```python
data['LUNG_CANCER'] = data['LUNG_CANCER'].map({'YES': 1, 'NO': 0})
```

```
/var/folders/g7/k8tb4tqx737cssc6482hmgb40000gn/T/ipykernel_1593/261
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pan
  data['LUNG_CANCER'] = data['LUNG_CANCER'].map({'YES': 1, 'NO': 0}
```

```python
data['LUNG_CANCER']
```

```
0      1
1      1
2      0
3      0
4      0
      ..
279    1
280    0
281    0
282    0
283    1
Name: LUNG_CANCER, Length: 276, dtype: int64
```

BONUS TIP:
  - Remove any trailing whitespace.
  - Convert the cols that are considered as "categorical" to type "category".
    - reduces memory usage.
    - allows more efficient data manipulation and plotting.

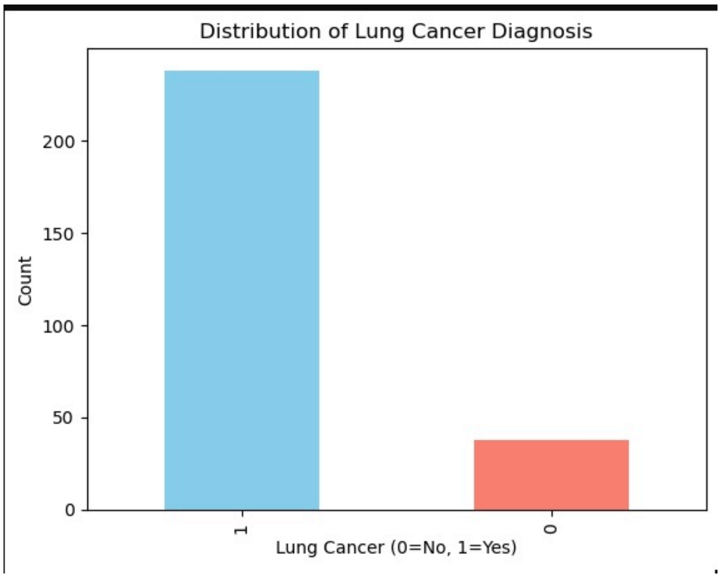
```
data.columns = data.columns.str.strip()
categorical_columns = [
    'GENDER', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY', 'PEER_PRESSURE',
    'CHRONIC DISEASE', 'FATIGUE', 'ALLERGY', 'WHEEZING', 'ALCOHOL CONSUMING',
    'COUGHING', 'SHORTNESS OF BREATH', 'SWALLOWING DIFFICULTY', 'CHEST PAIN'
]

for col in categorical_columns:
    data[col] = data[col].astype('category')
```

```
data.info()
```

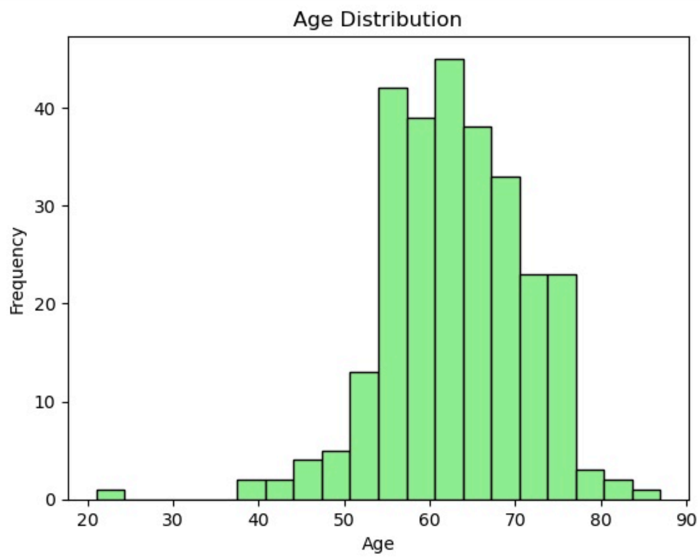
```
<class 'pandas.core.frame.DataFrame'>
Index: 276 entries, 0 to 283
Data columns (total 16 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   GENDER                 276 non-null    category
 1   AGE                    276 non-null    int64
 2   SMOKING                276 non-null    category
 3   YELLOW_FINGERS         276 non-null    category
 4   ANXIETY                276 non-null    category
 5   PEER_PRESSURE          276 non-null    category
 6   CHRONIC DISEASE        276 non-null    category
 7   FATIGUE                276 non-null    category
 8   ALLERGY                276 non-null    category
 9   WHEEZING               276 non-null    category
 10  ALCOHOL CONSUMING      276 non-null    category
 11  COUGHING               276 non-null    category
 12  SHORTNESS OF BREATH    276 non-null    category
 13  SWALLOWING DIFFICULTY  276 non-null    category
 14  CHEST PAIN             276 non-null    category
 15  LUNG_CANCER            276 non-null    int64
dtypes: category(14), int64(2)
memory usage: 11.9 KB
```

- **Quick Data Visualization**

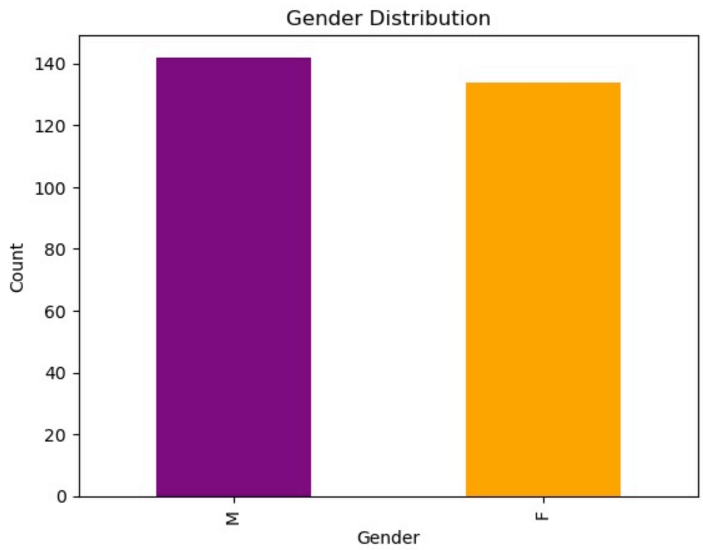
```
# Target variable
data['LUNG_CANCER'].value_counts().plot(kind='bar', color=['skyblue', 'salmon'])
plt.title('Distribution of Lung Cancer Diagnosis')
plt.xlabel('Lung Cancer (0=No, 1=Yes)')
plt.ylabel('Count')
plt.show()

# Age distribution
plt.hist(data['AGE'], bins=20, color='lightgreen', edgecolor='black')
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()

# Gender distribution
data['GENDER'].value_counts().plot(kind='bar', color=['purple', 'orange'])
plt.title('Gender Distribution')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()
```



- Strong imbalance of Cancer to no-cancer



- Age group is fairly normally distributed with most patients between 50-75 years old.



- Genders are fairly balanced.

- **Advance Data Inspection**
  - **Multivariate Outlier Detection**

```python
import numpy as np
from sklearn.ensemble import IsolationForest
data.columns = data.columns.str.strip()
# Correct: Create a copy of the original data
outlier_data = data.copy()

# Convert categorical columns to numeric codes
for col in categorical_columns:
    data[col] = data[col].astype('category')  # Convert to category FIRST
    print(f"{col} dtype: {data[col].dtype}")  # Verify conversion
    outlier_data = data.copy()
for col in categorical_columns:
    outlier_data[col] = outlier_data[col].cat.codes
```

```
GENDER dtype: category
SMOKING dtype: category
YELLOW_FINGERS dtype: category
ANXIETY dtype: category
PEER_PRESSURE dtype: category
CHRONIC DISEASE dtype: category
FATIGUE dtype: category
ALLERGY dtype: category
WHEEZING dtype: category
ALCOHOL CONSUMING dtype: category
COUGHING dtype: category
SHORTNESS OF BREATH dtype: category
SWALLOWING DIFFICULTY dtype: category
CHEST PAIN dtype: category
```

```python
# Verify conversion
print(outlier_data[categorical_columns].head())

# Detect outliers (exclude target variable)
iso = IsolationForest(contamination=0.05, random_state=42)
outliers = iso.fit_predict(outlier_data.drop('LUNG_CANCER', axis=1))

# Add outlier flags to original data
data['OUTLIER_FLAG'] = np.where(outliers == -1, 1, 0)

# Analyze outliers
outlier_profile = data[data['OUTLIER_FLAG'] == 1].describe().T
print("Outlier Summary:\n", outlier_profile[['mean', 'std', 'min', 'max']])
```

```
   GENDER  SMOKING  YELLOW_FINGERS  ANXIETY  PEER_PRESSURE  CHRONIC DISEASE
0       1        0               1        1              0                0
1       1        1               0        0              0                1
2       0        0               0        0              1                0
3       1        1               1        1              0                0
4       0        0               1        0              0                0

   FATIGUE  ALLERGY  WHEEZING  ALCOHOL CONSUMING  COUGHING  \
0        1        0         1                  1         1
1        1        1         0                  0         0
2        1        0         1                  0         1
3        0        0         0                  1         0
4        0        0         1                  0         1

   SHORTNESS OF BREATH  SWALLOWING DIFFICULTY  CHEST PAIN
0                    1                      1           1
1                    1                      1           1
2                    1                      0           1
3                    0                      1           1
4                    1                      0           0
```

- Outliers cluster around mean age 62 +/- 7.3 years
- Age Range: 47-77 years
- Age only continuous variable.
- Any outliers are flagged as 1.

	mean	std	min	max
AGE	62.642857	7.344356	47.0	77.0
LUNG_CANCER	0.642857	0.497245	0.0	1.0
OUTLIER_FLAG	1.000000	0.000000	1.0	1.0

- **Advance Numerical Analysis**

```
from scipy.stats import kendalltau

# Create symptom score dynamically from existing columns
symptoms = ['COUGHING', 'SHORTNESS OF BREATH', 'CHEST PAIN',
            'SWALLOWING DIFFICULTY', 'WHEEZING']
# Convert to numerical codes
for col in symptoms:
    data3[col] = data3[col].astype('category')  # Ensure categorical dtype
    data3[col] = data3[col].cat.codes.astype('int64')  # Force int64
# Create symptom score with int64 dtype
data3['SYMPTOM_SCORE'] = data3[symptoms].sum(axis=1).astype('int64')

# Ensure target is int64
data3['LUNG_CANCER'] = data3['LUNG_CANCER'].astype('int64')
```

```
int8_cols = data3.select_dtypes(include=['int8']).columns
data3[int8_cols] = data3[int8_cols].astype('int64')
```

```
# Analyze non-parametric correlations
num_features = ['AGE', 'SYMPTOM_SCORE']
target = data['LUNG_CANCER']

corr_results = []
for feat in num_features:
    tau, p_value = kendalltau(data[feat], target)
    corr_results.append({
        'Feature': feat,
        'Kendall Tau': tau,
        'p-value': p_value
    })

corr_df = pd.DataFrame(corr_results)
print("Non-Parametric Correlation:\n", corr_df)
```

```
Non-Parametric Correlation:
          Feature  Kendall Tau       p-value
0            AGE     0.080379  1.092943e-01
1  SYMPTOM_SCORE     0.339395  4.183125e-10
```

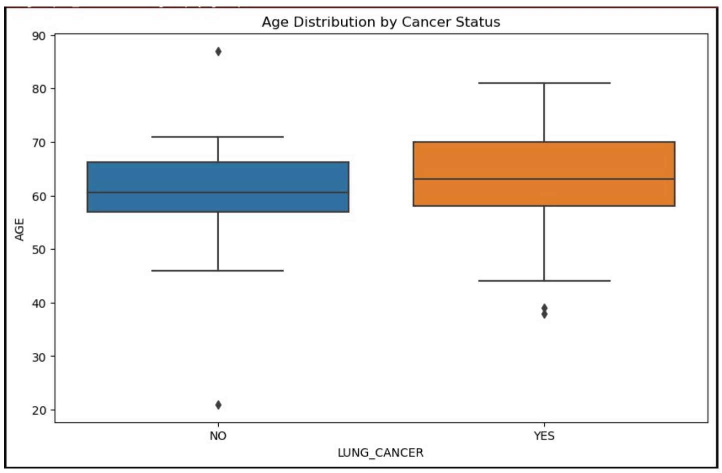
- Age vs Lung Cancer
  - Our Kendal Tau value of 0.08 shows a weak positive relationship
  - p = 0.109, thus it is not statistically sig.
- Symptom Score vs Lung Cancer
  - Kendal Tau = 0.34
    - moderate positive relationship
  - p = less than 4.18x^-10
    - highly statistically sig.
- For every 1 unit increase, there is 34% increased likelihood of lung cancer.
- Odd Ratio = 1 + tau / 1 - tau = 2.03
  - So 1 unit increase = 103% increase in **ODDS**

EXPLORATORY DATA ANALYSIS

- **Target Variable Analysis**

```
import seaborn as sns
# Class distribution with detailed metrics
target_dist = data3['LUNG_CANCER'].value_counts()
print(f"Class Balance:\n{target_dist}")
print(f"\nPositive Class Percentage: {target_dist[1]/len(data3)*100:.1f}%")
```

```
Class Balance:
LUNG_CANCER
1    238
0     38
Name: count, dtype: int64

Positive Class Percentage: 86.2%
```



- We have 238 cases (86.2%) that are pos for lung cancer.
- 39 cases (13.8%) are negative.
- From the boxplot
  - both groups have similar median ages. Ranges and outliers differ.
  - IQR is wider for pos cases.
  - Median Line: 50% of cancer patients are older than 62, while 50% of non-cancer patients are older than 56.

```
counts = data3['LUNG_CANCER'].value_counts()
percentages = counts / len(data3) * 100
print("Class distribution:\n", counts)
print("Class percentages:\n", percentages)
```

```
Class distribution:
 LUNG_CANCER
YES    238
NO      38
Name: count, dtype: int64
Class percentages:
 LUNG_CANCER
YES    86.231884
NO     13.768116
Name: count, dtype: float64
```

- 238 Pos case of LC (86%)
- 38 Neg case of LC (14%)

- When we look at class balances we can see a strong class imbalance exist.

```
# Compare mean ages
mean_ages = data3.groupby('LUNG_CANCER')['AGE'].mean()
print("Mean ages by group:\n", mean_ages)

# T-test for age difference
from scipy.stats import ttest_ind
ages_yes = data3[data3['LUNG_CANCER'] == 1]['AGE'].dropna()
ages_no = data3[data3['LUNG_CANCER'] == 0]['AGE'].dropna()
t_stat, p_val = ttest_ind(ages_yes, ages_no, equal_var=False)
print(f"T-test: t={t_stat:.2f}, p={p_val:.4f}")
```

```
Mean ages by group:
 LUNG_CANCER
0    60.684211
1    63.264706
Name: AGE, dtype: float64
T-test: t=1.55, p=0.1285
```

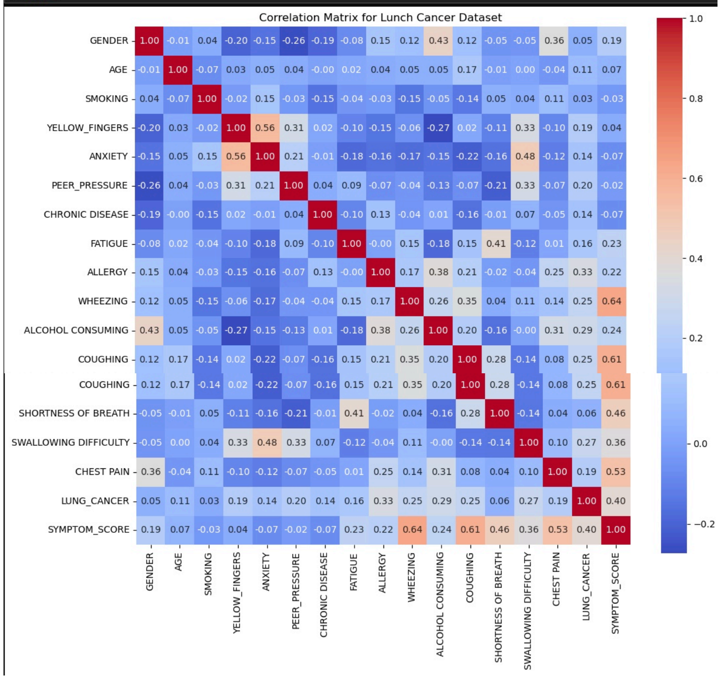
```
[40]: categorical_columns2 = ["GENDER","SMOKING","YELLOW_FINGERS","ANXIETY","PEER_PRESSURE",
                            "CHRONIC DISEASE", "FATIGUE", "ALLERGY","WHEEZING","ALCOHOL CONSUMING",
                            "COUGHING", "SHORTNESS OF BREATH","SWALLOWING DIFFICULTY", "CHEST PAIN"]

      for col in categorical_columns2:
          data3[col] = data3[col].astype("category").cat.codes

      corr_matrix = data3.corr(numeric_only=True)
      #print(corr_matrix["LUNG_CANCER"].sort_values(ascending= False))

[48]: #corr_matrix.info()

[44]: print(corr_matrix["LUNG_CANCER"].sort_values(ascending=False))

      LUNG_CANCER              1.000000
      SYMPTOM_SCORE            0.397805
      ALLERGY                  0.333552
      ALCOHOL CONSUMING        0.294422
      SWALLOWING DIFFICULTY    0.268940
      COUGHING                 0.253027
      WHEEZING                 0.249054
      PEER_PRESSURE            0.195086
      CHEST PAIN               0.194856
      YELLOW_FINGERS           0.189192
      FATIGUE                  0.160078
      ANXIETY                  0.144322
      CHRONIC DISEASE          0.143692
      AGE                      0.106305
      SHORTNESS OF BREATH      0.064407
      GENDER                   0.053666
      SMOKING                  0.034878
      Name: LUNG_CANCER, dtype: float64
```

- Patient without lung cancer have an avg age of 60.7
- Patient with lung cancer have an avg age of 63.26
- t-statistic = 1.55
- p-value = 0.1285
  - Age difference is not statistically sig.
  - Cannot reject the null hypothesis that the age distributions are the same
- This suggests that while lung cancer patients are slightly older on average in your dataset (63.3 vs 60.7 years), age alone is not a strong differentiating factor

- **Correlation Analysis**

- Allergy shows the 2nd highest correlation with lung cancer diagnosis (0.33)
- Smoking as a very weak correlation.
  - VERY SURPRISING!!!!
- SMOKING shows near-zero correlation (0.03) (NEED TO BE INVESTIGATED FURTHER)

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(12,10))
sns.heatmap(corr_matrix, annot=True, fmt=".2f", cmap="coolwarm", square=True)
plt.title("Correlation Matrix for Lunch Cancer Dataset")
plt.show()
```



- Dark Red = Strong Pos
- Dark Blue = Strong Neg

ANALYSIS RESULTS

- Class Distribution Analysis
  - Yes: 238 cases (86.2%)
  - No: 38 (13.8%)
  - Severe imbalance.
- Age Comparison Between Groups
  - Yes mean: 63.26 +/- 8.1 yrs
  - No mean: 60.68 +/- 9.8 yrs
- Welch's t-test
  - t= 1.55
  - p = 0.1285
    - Not Stat sig
- Symptom Score Relationship
  - Kendall's Tau
    - 0.34 and p = < 0.001
  - Strong pos association between symptom score and cancer likelihood.
- Feature Correlation Analysis
  - Point=Biserial Correlation
    - Symptom_score = 0.40
    - Allergy = 0.33
    - Alcohol consuming = 0.29
- Categorical Feature Analysis
  - Chi-Squre Test