Nanditha.V
160119733133

Data Science and Big Data Analytics

Experiment 4: Chi-Square Test, ANOVA, Correlation

Aim: To perform Chi-square test, ANOVA, Pearson Correlation, Correlation Heatmap.

Description: An Analysis of Variance (ANOVA) is a statistical test employed to compare twoor more means together, which are determined through the analysis of variance. One-way ANOVA tests are utilized to analyze differences between groups and determine if the differences are statistically significant.

One-way ANOVAs compare two or more independent group means, though in practice theyare most oftenly used when there are at least three independent groups.

In order to carry out an ANOVA on the Gapminder dataset, we'll need to transform some ofthe features, as these values in the dataset are continuous but ANOVA analyses are appropriate for situations where one variable is categorical and one variable is quantitative.

The Chi-Square test of independence is utilized when both explanatory and response variables are categorical. You likely also want to use the Chi-Square test when the explanatory variable is quantitative and the response variable is categorical, which you cando by dividing the explanatory variable into categories.

The Chi-Square test of independence is a statistical test used to analyze how significant a relationship between two categorical variables is. When a Chi-Square test is run, every category in one variable has its frequency compared against the second variable's categories. This means that the data can be displayed as a frequency table, where the rowsrepresent the independent variables and the columns represent the dependent variables.

The Pearson Correlation test is used to analyze the strength of a relationship between twoprovided variables, both quantitative in nature. The value, or strength of the Pearson correlation, will be between +1 and -1.

A correlation of 1 indicates a perfect association between the variables, and the correlation is either positive or negative. Correlation coefficients near 0 indicate very weak, almost non-existent, correlations. While there are other ways of measuring correlations between two variables, such as Spearman Correlation or Kendall Rank Correlation, Pearson correlation is probably the most commonly used correlational test.

STEP – 1:

Import Modules

```
import statsmodels.formula.api as smf

import statsmodels.stats.multicomp as multi
import scipy
```

```
from scipy.stats import pearsonr
import pandas as pd
from seaborn import regplot
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
```

HeatMap

CODE:

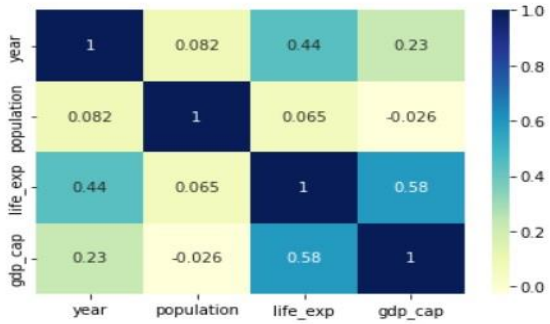
```
import matplotlib.pyplot as mp
import pandas as pd
import seaborn as sb

# import file with data
data = pd.read_csv("countrylifgdp.csv")
print(data.info())


print(data.corr())

# plotting correlation heatmap
dataplot = sb.heatmap(data.corr(), cmap="YlGnBu", annot=True)

# displaying heatmap
mp.show()
```

OUTPUT:

Nanditha.V
160119733133

```
In [4]: # prints data that will be plotted
        # columns shown here are selected by corr() since
        # they are ideal for the plot
        print(data.corr())

        # plotting correlation heatmap
        dataplot = sb.heatmap(data.corr(), cmap="YlGnBu", annot=True)

        # displaying heatmap
        mp.show()
```

```
                year  population  life_exp   gdp_cap
year        1.000000    0.082308  0.435611  0.227318
population  0.082308    1.000000  0.064955 -0.025600
life_exp    0.435611    0.064955  1.000000  0.583706
gdp_cap     0.227318   -0.025600  0.583706  1.000000
```



Output Analysis:

Life_exp and gdp_cap have the highest positive correlation, population and gdp_cap has negative correlation.

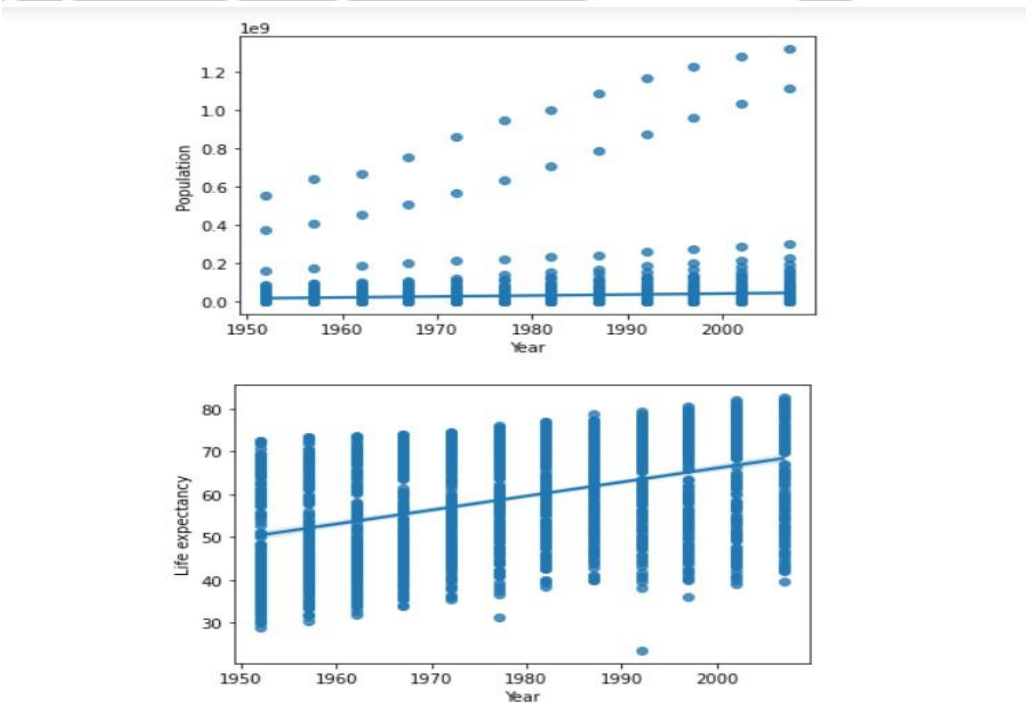
2) Pearson Correlation

CODE:

```
df_clean = data.dropna()

def plt_regression(x, y, data, label_1, label_2):
    reg_plot = regplot(x=x, y=y, fit_reg=True,
    data=data)plt.xlabel(label_1)
    plt.ylabel(label_2)
    plt.show()

plt_regression('year', 'population', df_clean, 'Year', 'Population')
```

```
plt_regression('year', 'life_exp', df_clean, 'Year', 'Life expectancy')

print('Assoc. - life expectancy and internet use rate')
print(pearsonr(df_clean['year'], df_clean['population']))

print('Assoc. - between employment rate and internet use rate')
print(pearsonr(df_clean['year'], df_clean['life_exp']))
```

OUTPUT:




```
Assoc. - life expectancy and internet use rate
(0.08230807783830922, 0.0006716006496141012)
Assoc. - between employment rate and internet use rate
(0.4356112240540735, 7.54679462559958e-80)
```

Output Analysis:

As we saw in the heatmap year, population and year, life_Exp have positive correlation in the same way pearson graph also has a positive slope.

3) ANOVA

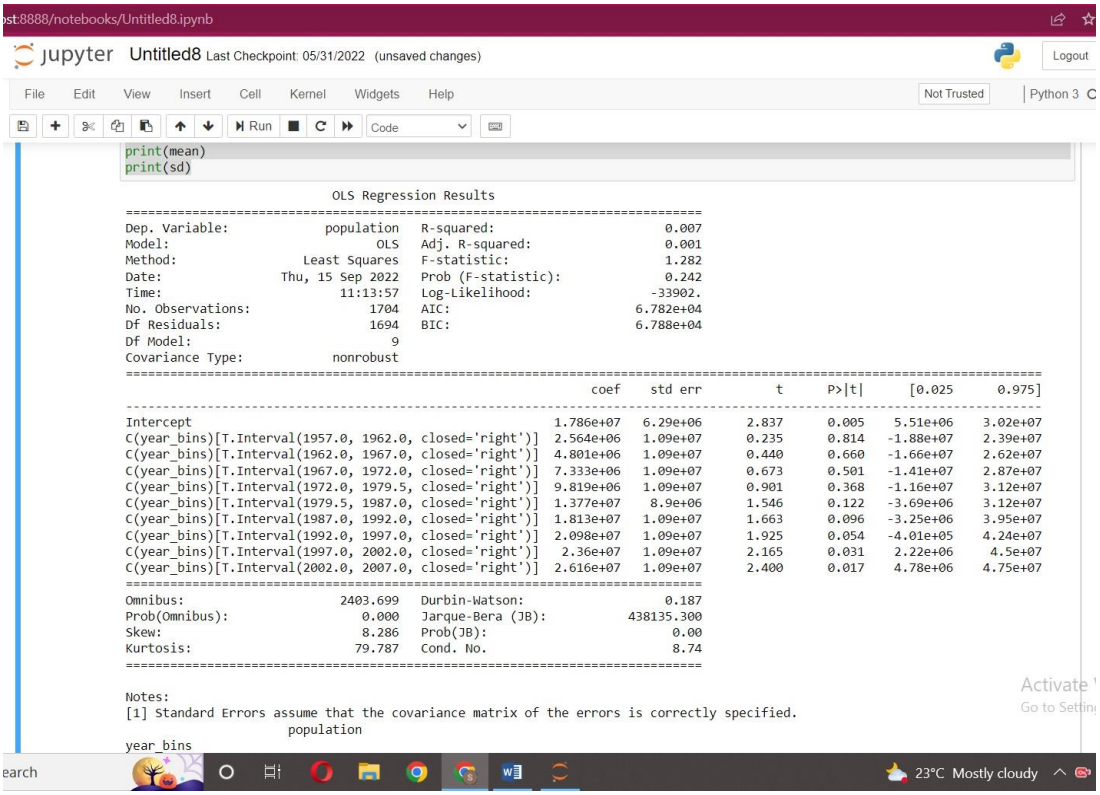
Nanditha.V
160119733133

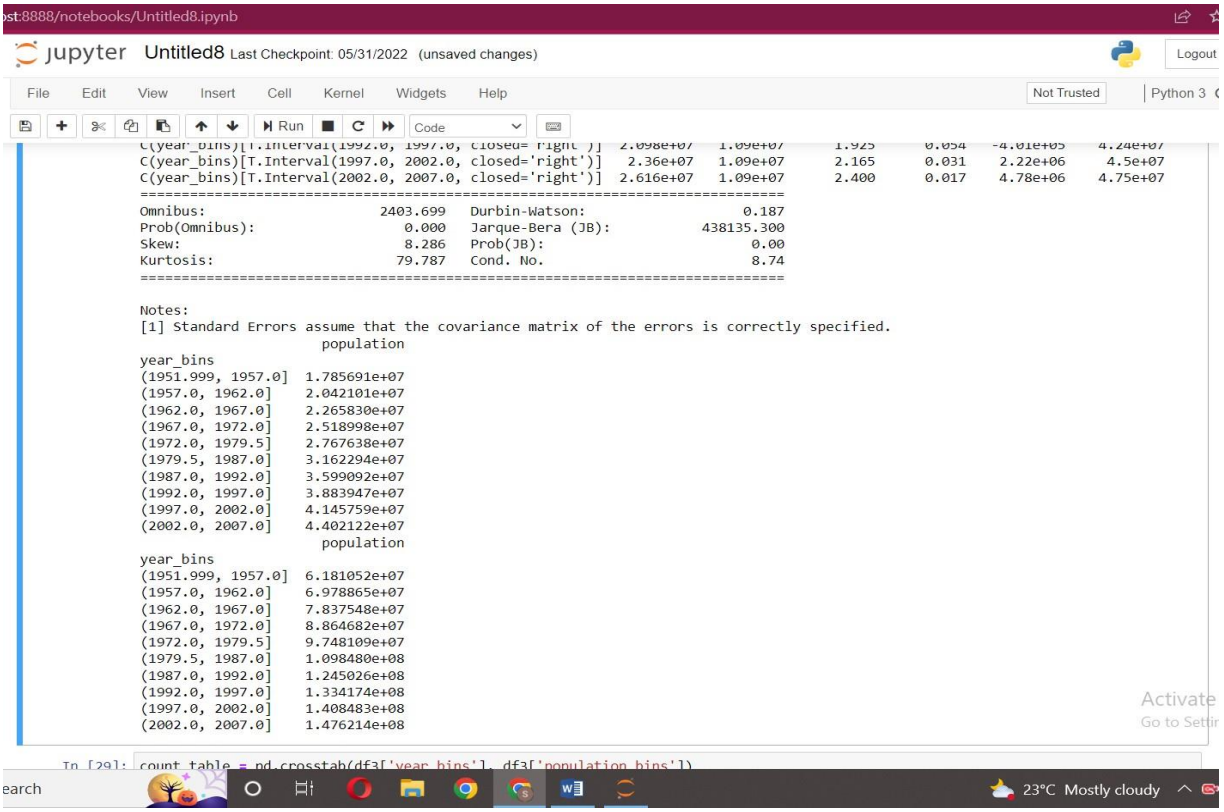
CODE:

```
def bin(dataframe, cols):
        # Create new columns that store the binned data
   for col in cols:
      new_col_name = "{}_bins".format(col)
      dataframe[new_col_name] = pd.qcut(dataframe[col], 10)

df3 = data.copy()
cols = ['year']
cols1 = ['population']
norm_cols = ['population' , 'gdp_cap' , 'life_exp']
# This creates new columns filled with the binned column data
bin(df3, cols)
bin(df3, norm_cols)
bin(df3, cols1)

anova_df = df3[['year_bins', 'population']].dropna()

relate_df = df3[['year_bins', 'population']]

anova = smf.ols(formula='population ~ C(year_bins)', data=anova_df).fit()

print(anova.summary())

# We may also want to check the mean and standard deviation for the groups
mean = relate_df.groupby("year_bins").mean()
sd = relate_df.groupby("year_bins").std()
print(mean)
print(sd)
```

Nanditha.V
160119733133

OUTPUT:





OUTPUT Analysis:

P-Value is 0.242 > 0.05 [Level of Significance] which implies that we don't have a significantrelationship between population and year.

4) Chi Square Test

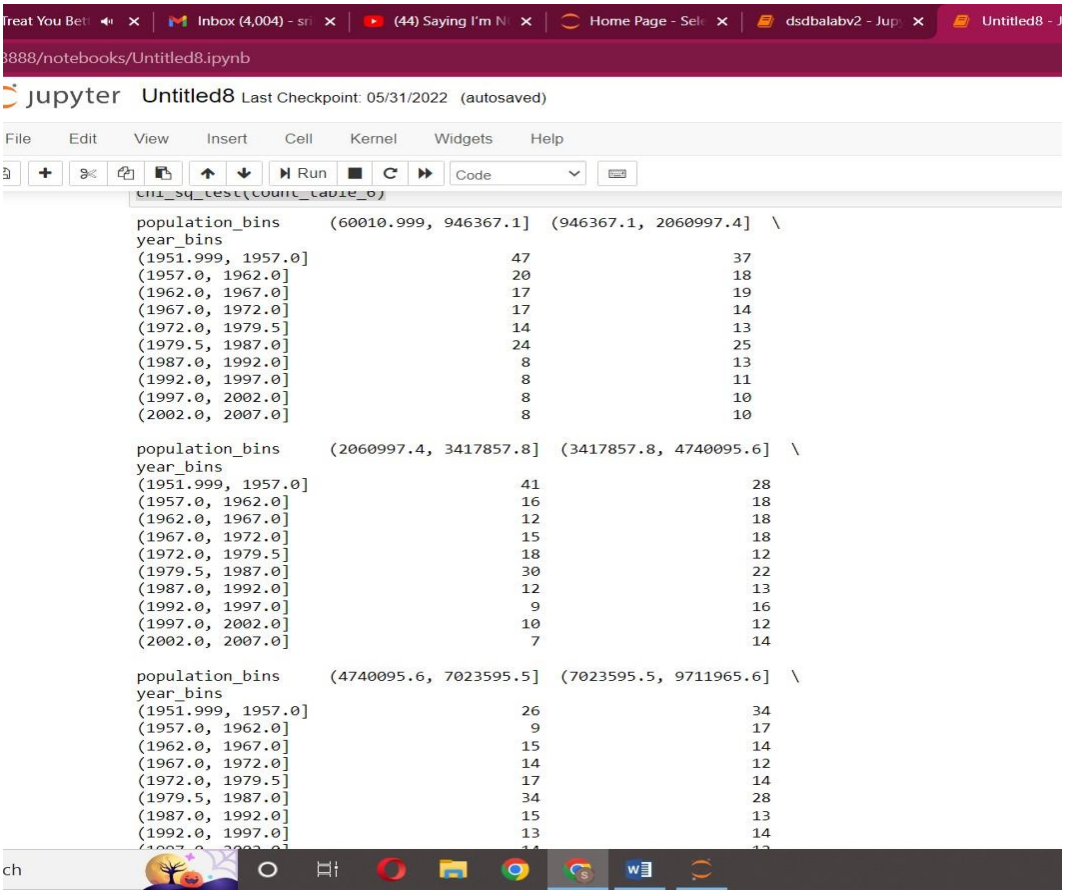
CODE:

```
count_table = pd.crosstab(df3['year_bins'], df3['population_bins'])

print(count_table)


def chi_sq_test(table):

    print("Results for:")

    print(str(table))


    # Get column percentages
```

```
    col_sum = table.sum(axis=0)
    col_percents = table/col_sum
    print(col_percents)

    chi_square = scipy.stats.chi2_contingency(table)
    print("Chi-square value, p-value, expected_counts")
    print(chi_square)

    print()

print("Initial Chi-
square:")
chi_sq_test(count_table)
print(" ")

chi_sq_test(count_table_3)
chi_sq_test(count_table_4)
chi_sq_test(count_table_5)
chi_sq_test(count_table_6)
```

OUTPUT:

Nanditha.V
160119733133



population_bins	(60010.999, 946367.1]	(946367.1, 2060997.4]
year_bins		
(1951.999, 1957.0]	47	37
(1957.0, 1962.0]	20	18
(1962.0, 1967.0]	17	19
(1967.0, 1972.0]	17	14
(1972.0, 1979.5]	14	13
(1979.5, 1987.0]	24	25
(1987.0, 1992.0]	8	13
(1992.0, 1997.0]	8	11
(1997.0, 2002.0]	8	10
(2002.0, 2007.0]	8	10

population_bins	(2060997.4, 3417857.8]	(3417857.8, 4740095.6]
year_bins		
(1951.999, 1957.0]	41	28
(1957.0, 1962.0]	16	18
(1962.0, 1967.0]	12	18
(1967.0, 1972.0]	15	18
(1972.0, 1979.5]	18	12
(1979.5, 1987.0]	30	22
(1987.0, 1992.0]	12	13
(1992.0, 1997.0]	9	16
(1997.0, 2002.0]	10	12
(2002.0, 2007.0]	7	14

population_bins	(4740095.6, 7023595.5]	(7023595.5, 9711965.6]
year_bins		
(1951.999, 1957.0]	26	34
(1957.0, 1962.0]	9	17
(1962.0, 1967.0]	15	14
(1967.0, 1972.0]	14	12
(1972.0, 1979.5]	17	14
(1979.5, 1987.0]	34	28
(1987.0, 1992.0]	15	13
(1992.0, 1997.0]	13	14