Hypothesis Testing

Wittawat Muangkot

2022-04-12

this paper will analyze Hoston_housing dataset and do the Hypothesis testing to answer question to find insight from dataset

- T-test
- ANOVA
- Chi- square
- correlation
- Regression

Import nescessary package

```
import pandas as pd
import numpy as np
import scipy.stats
import seaborn as sns
import matplotlib.pyplot as plt
import statsmodels.api as sm

from IPython import get_ipython
ipy = get_ipython()
if ipy is not None:
  ipy.run_line_magic('matplotlib','inline')


import warnings
warnings.filterwarnings('ignore')
```

Collection data from sklearn.dataset

```
from sklearn.datasets import load_boston
boston = load_boston()
df=pd.DataFrame(boston.data)
df.columns=boston.feature_names
df['MEDV']=boston.target
df.head()
```

```
##         CRIM    ZN  INDUS  CHAS    NOX  ...    TAX  PTRATIO       B  LSTAT  MEDV
## 0  0.00632  18.0   2.31   0.0  0.538  ...  296.0     15.3  396.90   4.98  24.0
## 1  0.02731   0.0   7.07   0.0  0.469  ...  242.0     17.8  396.90   9.14  21.6
## 2  0.02729   0.0   7.07   0.0  0.469  ...  242.0     17.8  392.83   4.03  34.7
## 3  0.03237   0.0   2.18   0.0  0.458  ...  222.0     18.7  394.63   2.94  33.4
## 4  0.06905   0.0   2.18   0.0  0.458  ...  222.0     18.7  396.90   5.33  36.2
##
## [5 rows x 14 columns]
```

Dataset Description

The following describes the dataset variables:

CRIM - per capita crime rate by town

ZN - proportion of residential land zoned for lots over 25,000 sq.ft.

INDUS - proportion of non-retail business acres per town.

CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)

NOX - nitric oxides concentration (parts per 10 million)

RM - average number of rooms per dwelling

AGE - proportion of owner-occupied units built prior to 1940

DIS - weighted distances to five Boston employment centres

RAD - index of accessibility to radial highways

TAX - full-value property-tax rate per $10,000

PTRATIO - pupil-teacher ratio by town

B - 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town

LSTAT - % lower status of the population

MEDV - Median value of owner-occupied homes in $1000's

Explore Data

```
df.info()
```

```
## <class 'pandas.core.frame.DataFrame'>
## RangeIndex: 506 entries, 0 to 505
## Data columns (total 14 columns):
##  #   Column   Non-Null Count  Dtype
## ---  ------   --------------  -----
##  0   CRIM     506 non-null    float64
##  1   ZN       506 non-null    float64
##  2   INDUS    506 non-null    float64
##  3   CHAS     506 non-null    float64
##  4   NOX      506 non-null    float64
##  5   RM       506 non-null    float64
##  6   AGE      506 non-null    float64
##  7   DIS      506 non-null    float64
```

```
##  8   RAD      506 non-null    float64
##  9   TAX      506 non-null    float64
##  10  PTRATIO  506 non-null    float64
##  11  B        506 non-null    float64
##  12  LSTAT    506 non-null    float64
##  13  MEDV     506 non-null    float64
## dtypes: float64(14)
## memory usage: 55.5 KB
```

Show basic Descriptive Statistics for each variable

As the columns "CHAS" is a binary value, so before we start vistaulization and find insight from this dataset, we add labels to each binary variable

```
df['CHAS']=np.where(df['CHAS']==1,'bounds river','otherwise')
df.describe(include='all')
```

```
##               CRIM          ZN       INDUS  ...           B       LSTAT        MEDV
## count   506.000000  506.000000  506.000000  ...  506.000000  506.000000  506.000000
## unique         NaN         NaN         NaN  ...         NaN         NaN         NaN
## top            NaN         NaN         NaN  ...         NaN         NaN         NaN
## freq           NaN         NaN         NaN  ...         NaN         NaN         NaN
## mean      3.613524   11.363636   11.136779  ...  356.674032   12.653063   22.532806
## std       8.601545   23.322453    6.860353  ...   91.294864    7.141062    9.197104
## min       0.006320    0.000000    0.460000  ...    0.320000    1.730000    5.000000
## 25%       0.082045    0.000000    5.190000  ...  375.377500    6.950000   17.025000
## 50%       0.256510    0.000000    9.690000  ...  391.440000   11.360000   21.200000
## 75%       3.677083   12.500000   18.100000  ...  396.225000   16.955000   25.000000
## max      88.976200  100.000000   27.740000  ...  396.900000   37.970000   50.000000
##
## [11 rows x 14 columns]
```

```
df['CHAS'].value_counts()
```

```
## otherwise       471
## bounds river     35
## Name: CHAS, dtype: int64
```

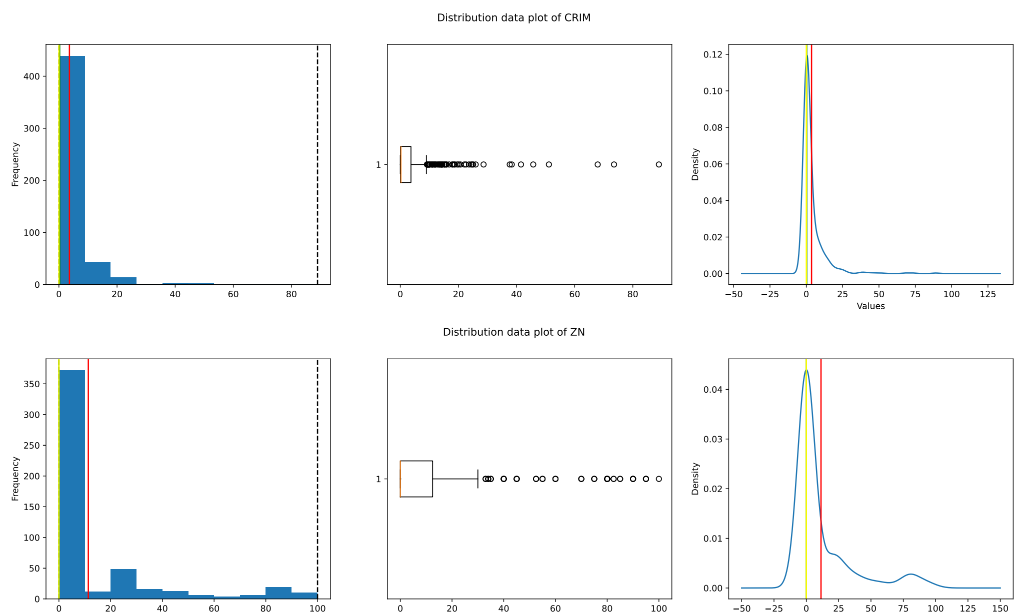
As we quick look at dataset, combline with 13 numeric data types with some columns with skew character and 1 category columns with 2 unique values. So we need to deep explore dataset by visualization to check data distribution and outliner

Data visualization

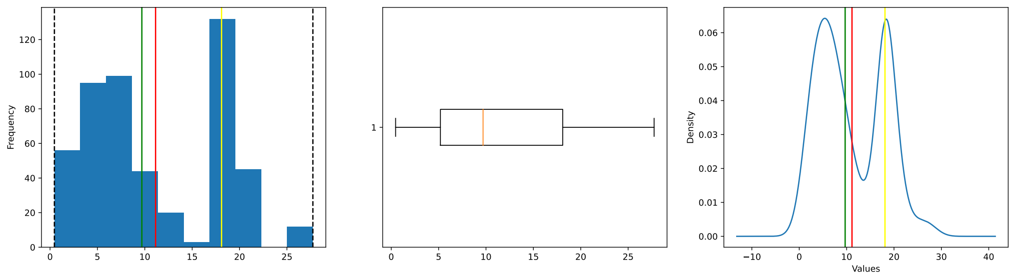
```
def distribution(col):
    for i in col:

        min_val=df[i].min()
        max_val=df[i].max()
```

```
        mean_val=df[i].mean()
        mid_val=df[i].median()
        mode_val=df[i].mode()[0]
        fig,ax=plt.subplots(1,3,figsize=(20,5))
        ax[0].hist(df[i])
        ax[0].axvline(min_val,color='black',linestyle='--')
        ax[0].axvline(max_val,color='black',linestyle='--')
        ax[0].axvline(mean_val,color='red')
        ax[0].axvline(mid_val,color='g')
        ax[0].axvline(mode_val,color='yellow')
        ax[0].set_ylabel('Frequency')

        ax[1].boxplot(df[i],vert=False)
        ax[2]=df[i].plot(kind='kde')
        ax[2].axvline(mean_val,color='red')
        ax[2].axvline(mid_val,color='g')
        ax[2].axvline(mode_val,color='yellow')
        ax[2].set_xlabel('Values')
        fig.suptitle('Distribution data plot of ' + i)

        plt.show()
```

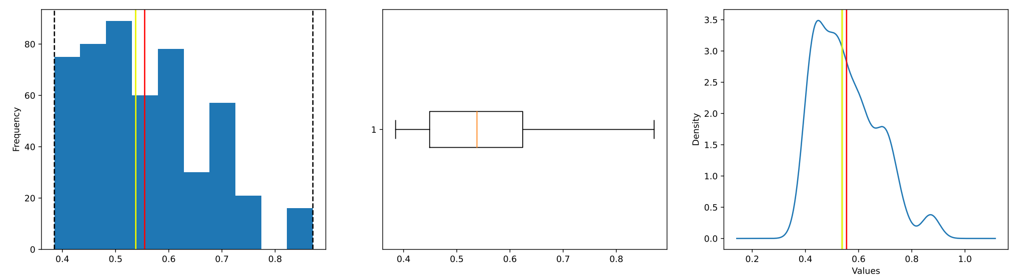
```
col_num=[x for x in df.columns
         if df[x].dtypes in ['int','float']]
distribution(col_num)
```



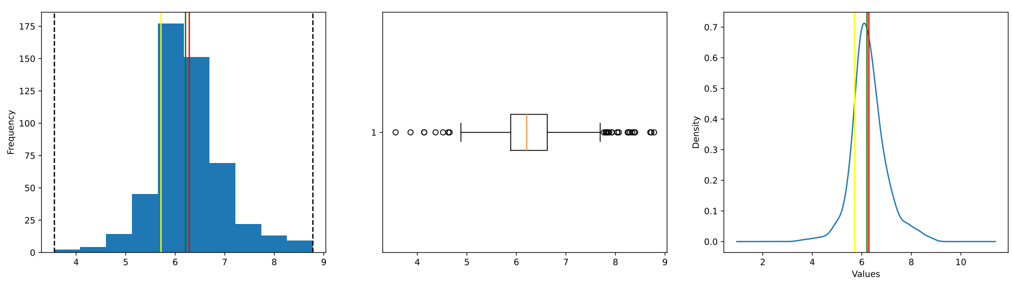
Distribution data plot of INDUS



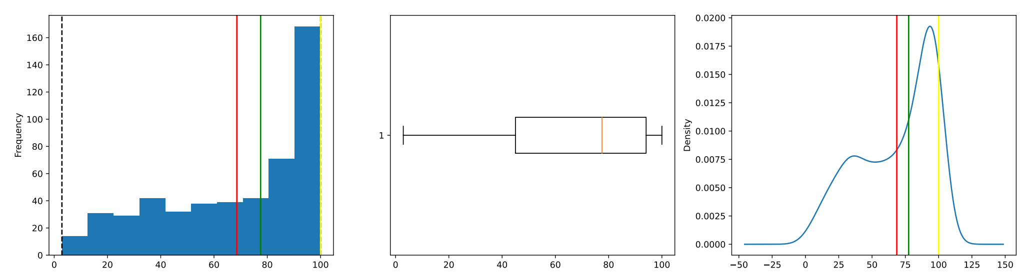
Distribution data plot of NOX



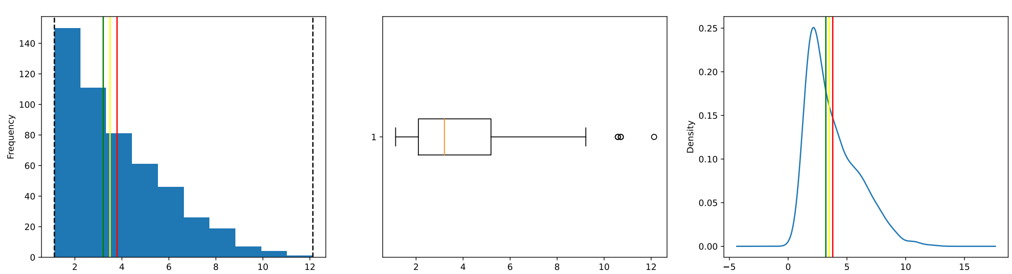
Distribution data plot of RM



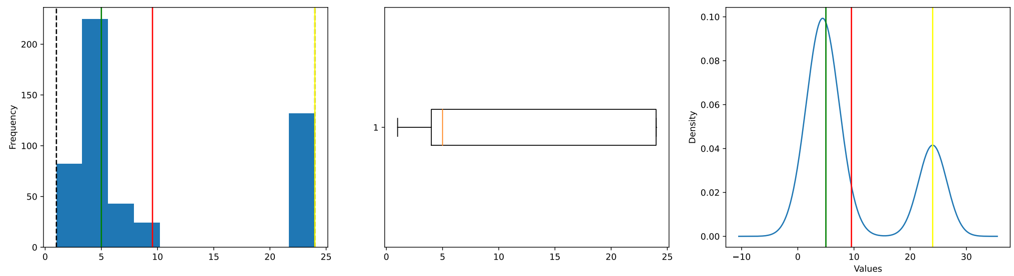
Distribution data plot of AGE



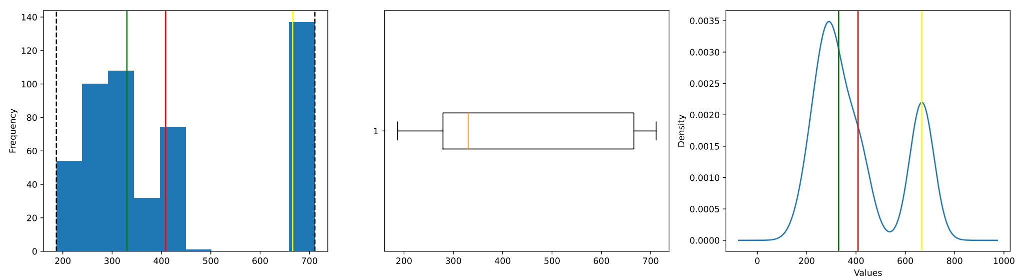
Distribution data plot of DIS



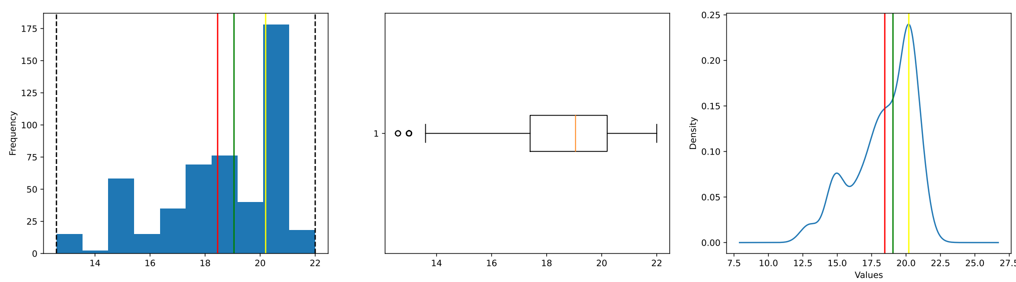
Distribution data plot of RAD



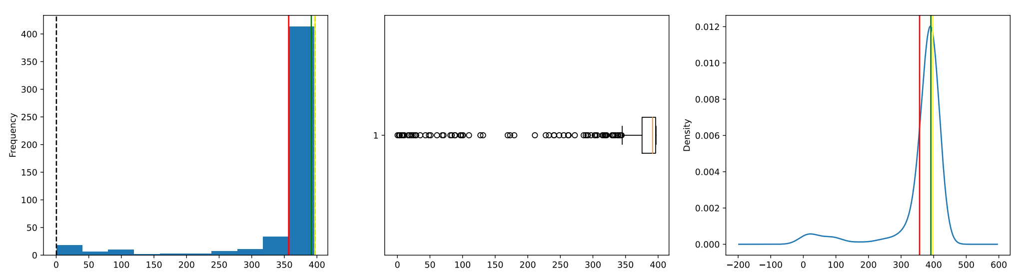
Distribution data plot of TAX



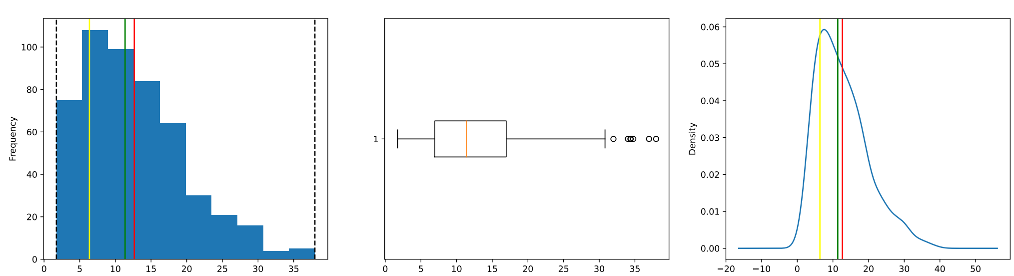
Distribution data plot of PTRATIO

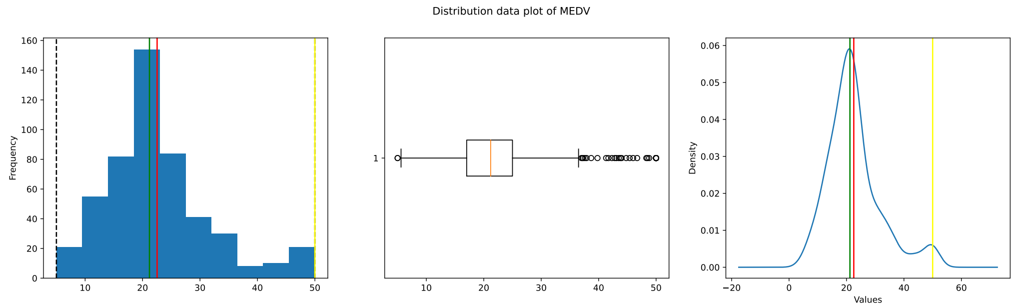


Distribution data plot of B



Distribution data plot of LSTAT





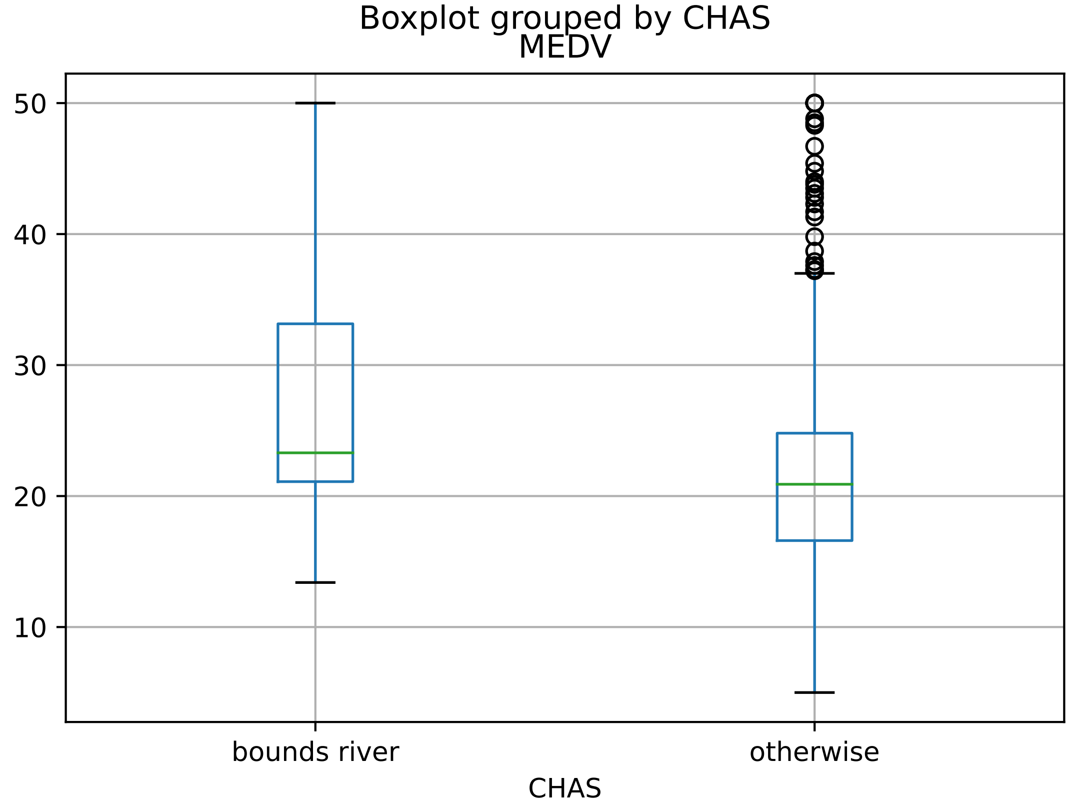
As we use visualization notice that some columns has skew and outliner, so it more effect to mean values. We need to do some precess in feature engineer like Transform, Discrete value precess before use this dataset to build model

Hypothesis Testing

Let start with process to answer question by do the hypothesis testing and choosing a test statistic (t-test, ANOVA, etc)

As the dataset has 1 category columns, so we want to know that in each type of columns "CHAS" has any difference average price of house "MEDV" ? we will use the T-test method:

```
df.boxplot(by='CHAS',column='MEDV')
plt.show()
```



State the hypothesis - H_0: mean_1 = mean_2 (“there is no difference in Median value of owner-occupied homes between bounds Charles River and otherwise”) - H_1: mean_1 != mean_2 (“there is a difference in Median value of owner-occupied homes between bounds Charles River and otherwise”) - alpha value 0.05

```
scipy.stats.levene(df[df['CHAS']=='bounds river']['MEDV'],
            df[df['CHAS']=='otherwise']['MEDV'], center='mean')
```

```
## LeveneResult(statistic=8.75190489604598, pvalue=0.003238119367639829)
```

since the p-value is smaller than 0.05 we can assume not equality of variance and we use the equal_var parameter as False

```
scipy.stats.ttest_ind(df[df['CHAS']=='bounds river']['MEDV'],
                     df[df['CHAS']=='otherwise']['MEDV'],equal_var=False)
```

```
## Ttest_indResult(statistic=3.113291312794837, pvalue=0.003567170098137517)
```

Conclusion:

since the P-value is smaller than alpha 0.05 we reject the null hypothesis and there is enough proof that there is the significant difference of Median value of owner-occupied homes base on Charles River location

Discretisation

we will process of transforming continuous variables into discrete variables by creating a set of contiguous intervals that span the range of the variable’s values into 3 unique high medium low to check the average values of each level are same.

```
col_continue= [x for x in df.columns
              if x != 'MEDV' and df[x].dtypes in ['int','float'] and df[x].nunique()>20]
col_continue
```

```
## ['CRIM', 'ZN', 'INDUS', 'NOX', 'RM', 'AGE', 'DIS', 'TAX', 'PTRATIO', 'B', 'LSTAT']
```

We will use columns ‘CRIM’,‘NOX’,‘AGE’,‘TAX’ to discrete data into high medium low category

```
col_list=['CRIM','NOX','AGE','TAX']
for i in col_list:
    df['Group_'+str(i)]=pd.cut(df[i],bins=3,labels=['low','medium','high'],ordered=True)
```

So let Answer the question

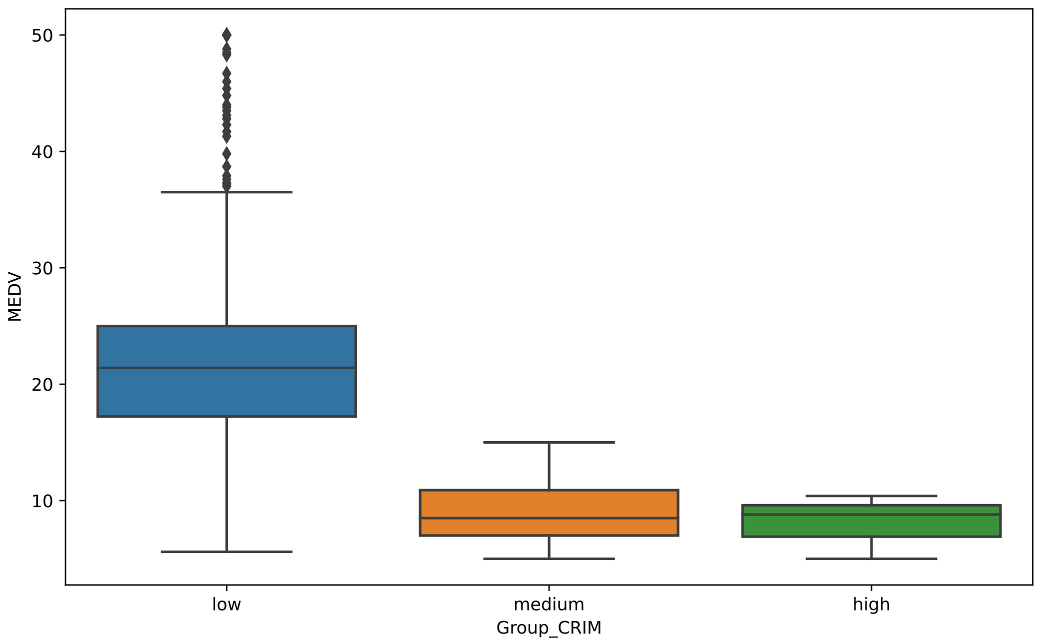
- the level of crime rate has effect to average price of house or not ?
- the level of nitric oxides concentration has effect to average price of house or not ?
- the house’s age has effect to average price of house or not ?
- the level of full-value property-tax rate has effect to average price of house or not ?

first start with descriptive statistic for each variable and visualize boxplot by each level of category ‘low’,‘medium’,‘high’

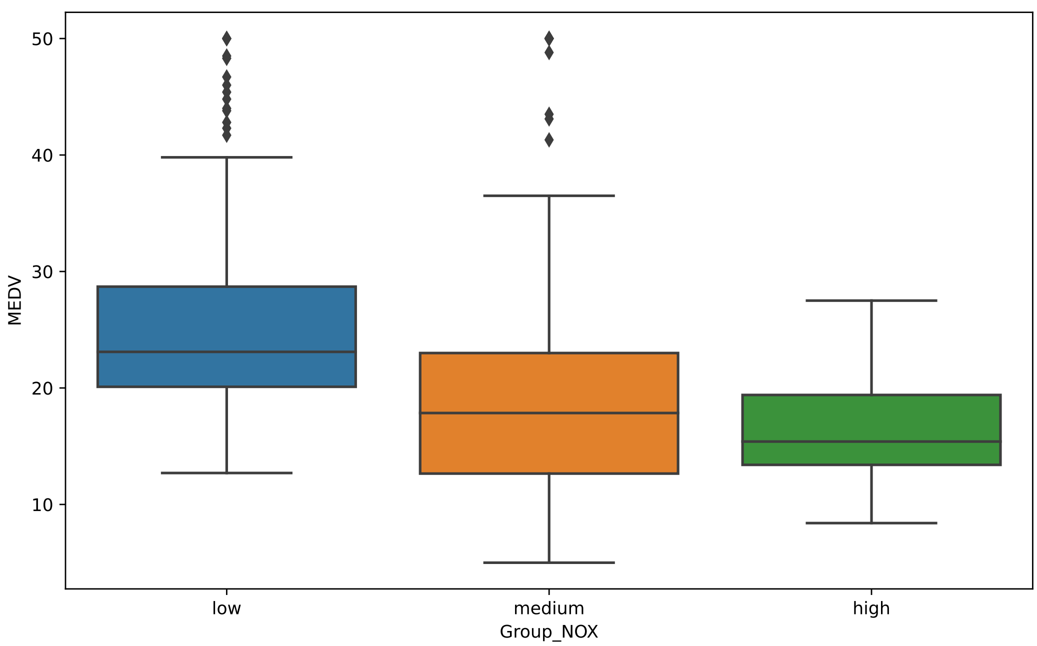
```python
group_col=['Group_'+str(x) for x in col_list]


for i in group_col:
    a=df.groupby(i).agg({'MEDV':['mean','std','var','min','max']}).reset_index()
    print(a)
```

```
##   Group_CRIM       MEDV
##                    mean       std        var  min   max
## 0        low  22.753012  9.094994  82.718914  5.6  50.0
## 1     medium   9.280000  3.855775  14.867000  5.0  15.0
## 2       high   8.066667  2.773686   7.693333  5.0  10.4
##   Group_NOX       MEDV
##                   mean        std         var   min   max
## 0       low  25.143345   7.585810   57.544519  12.7  50.0
## 1    medium  20.065789  11.351985  128.867564   5.0  50.0
## 2      high  16.140984   4.163827   17.337459   8.4  27.5
##   Group_AGE       MEDV
##                   mean       std        var   min   max
## 0       low  27.775824  7.638198  58.342076  17.1  50.0
## 1    medium  25.140336  7.142168  51.010563  10.2  50.0
## 2      high  19.872635  9.395453  88.274537   5.0  50.0
##   Group_TAX       MEDV
##                   mean       std        var   min   max
## 0       low  25.798168  8.243333  67.952533  11.9  50.0
## 1    medium  22.181250  8.084661  65.361750  11.8  50.0
## 2      high  16.272263  8.459008  71.554813   5.0  50.0
```

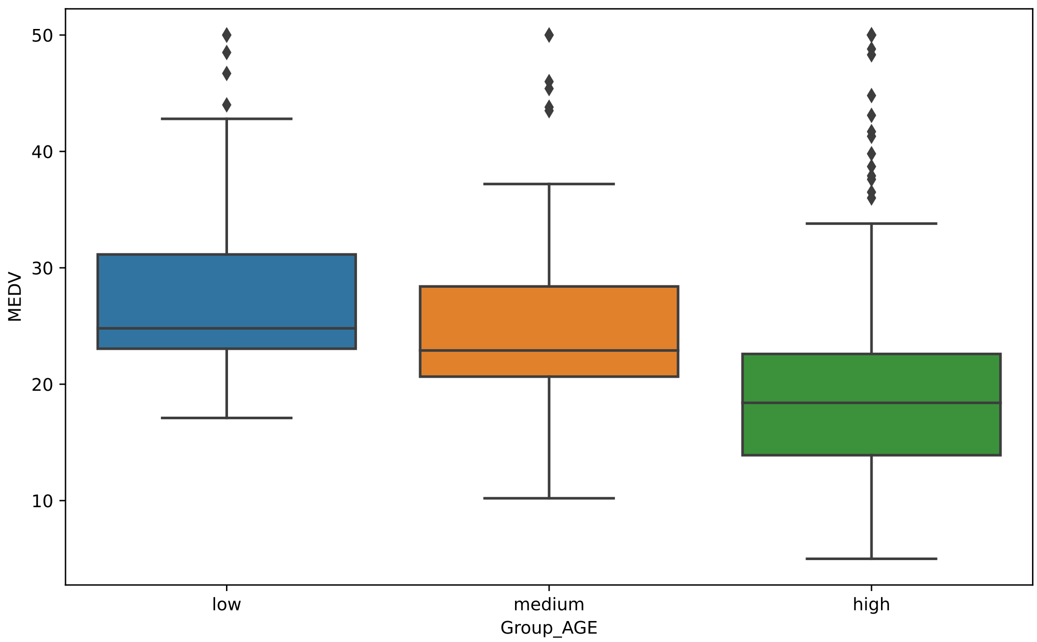
```python
for i in group_col:
    fig,ax=plt.subplots(figsize=(10,6))
    sns.boxplot(x=df[i],y=df['MEDV'],order=['low','medium','high'])
    plt.show()
```

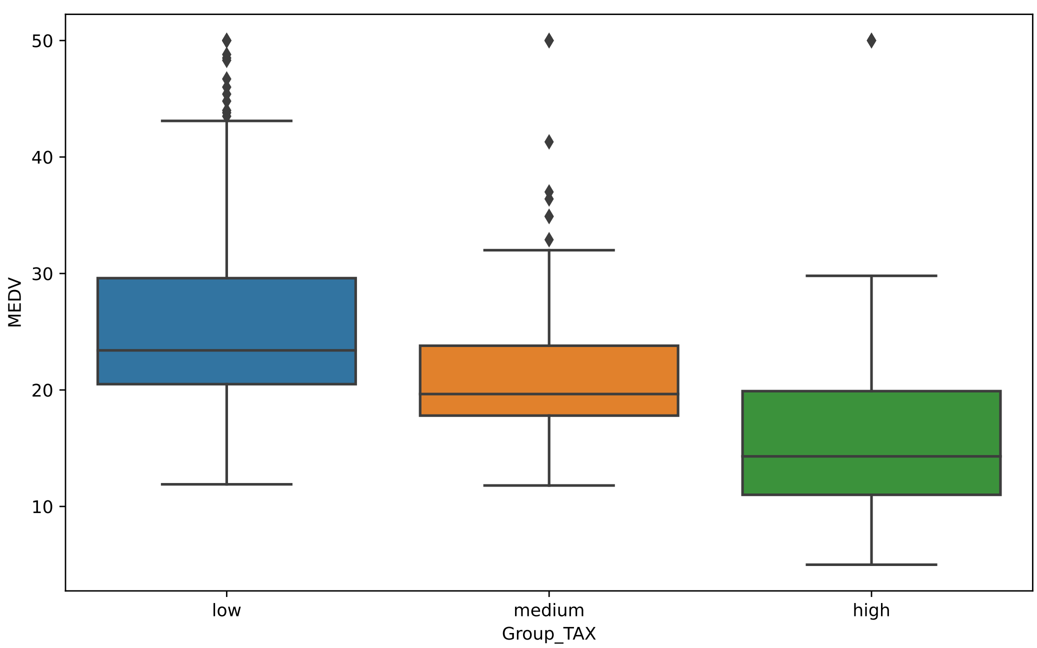


50
40
30
20
10
MEDV
low
medium
high
Group_CRIM



50
40
30
20
10
MEDV
low
medium
high
Group_NOX


50
40
30
20
10
MEDV
low
medium
high
Group_AGE


50
40
30
20
10
MEDV
low
medium
high
Group_TAX

As we see the above chart, notice that as high level of each variable the average price of house has the significant low compare to low level and medium level to confirm this assumption we will do test the Anova as there are more than 2 variable

‘

```
scipy.stats.levene(df[df['Group_CRIM']=='low']['MEDV'],
                   df[df['Group_CRIM']=='medium']['MEDV'],
                   df[df['Group_CRIM']=='high']['MEDV'],center='mean')
```

```
## LeveneResult(statistic=1.6108459729525946, pvalue=0.20074713497878735)
```

```
scipy.stats.f_oneway(df[df['Group_CRIM']=='low']['MEDV'],
                   df[df['Group_CRIM']=='medium']['MEDV'],
                   df[df['Group_CRIM']=='high']['MEDV'])
```

```
## F_onewayResult(statistic=9.343680555214876, pvalue=0.00010367020638159686)
```

Conclusion:

Since the p-value is less than 0.05, we will reject the null hypothesis as there is significant evidence that at least one of the means differ.

```
scipy.stats.levene(df[df['Group_NOX']=='low']['MEDV'],
                   df[df['Group_NOX']=='medium']['MEDV'],
                   df[df['Group_NOX']=='high']['MEDV'],center='mean')
```

```
## LeveneResult(statistic=16.64826257143882, pvalue=9.976886034704474e-08)
```

```
scipy.stats.f_oneway(df[df['Group_NOX']=='low']['MEDV'],
                   df[df['Group_NOX']=='medium']['MEDV'],
                   df[df['Group_NOX']=='high']['MEDV'])
```

```
## F_onewayResult(statistic=36.502695182504134, pvalue=1.574242045413178e-15)
```

Conclusion:

Since the p-value is less than 0.05, we will reject the null hypothesis as there is significant evidence that at least one of the means differ.

```
scipy.stats.levene(df[df['Group_AGE']=='low']['MEDV'],
                   df[df['Group_AGE']=='medium']['MEDV'],
                   df[df['Group_AGE']=='high']['MEDV'],center='mean')
```

```
## LeveneResult(statistic=1.7908490654674218, pvalue=0.16788045204452978)
```

```
scipy.stats.f_oneway(df[df['Group_AGE']=='low']['MEDV'],
                   df[df['Group_AGE']=='medium']['MEDV'],
                   df[df['Group_AGE']=='high']['MEDV'])
```

```
## F_onewayResult(statistic=36.434981466845564, pvalue=1.6701480198809046e-15)
```

Conclusion:

Since the p-value is less than 0.05, we will reject the null hypothesis as there is significant evidence that at least one of the means differ.

```
scipy.stats.levene(df[df['Group_TAX']=='low']['MEDV'],
                   df[df['Group_TAX']=='medium']['MEDV'],
                   df[df['Group_TAX']=='high']['MEDV'],center='mean')
```

```
## LeveneResult(statistic=0.8889940853260171, pvalue=0.41171390167059463)
```

```
scipy.stats.f_oneway(df[df['Group_TAX']=='low']['MEDV'],
                   df[df['Group_TAX']=='medium']['MEDV'],
                   df[df['Group_TAX']=='high']['MEDV'])
```

```
## F_onewayResult(statistic=60.58390673380421, pvalue=2.667188983508408e-24)
```

Conclusion:

Since the p-value is less than 0.05, we will reject the null hypothesis as there is significant evidence that at least one of the means differ.

Chi-square

```
cross_table=pd.crosstab(df['CHAS'],df['Group_AGE'])
cross_table
```

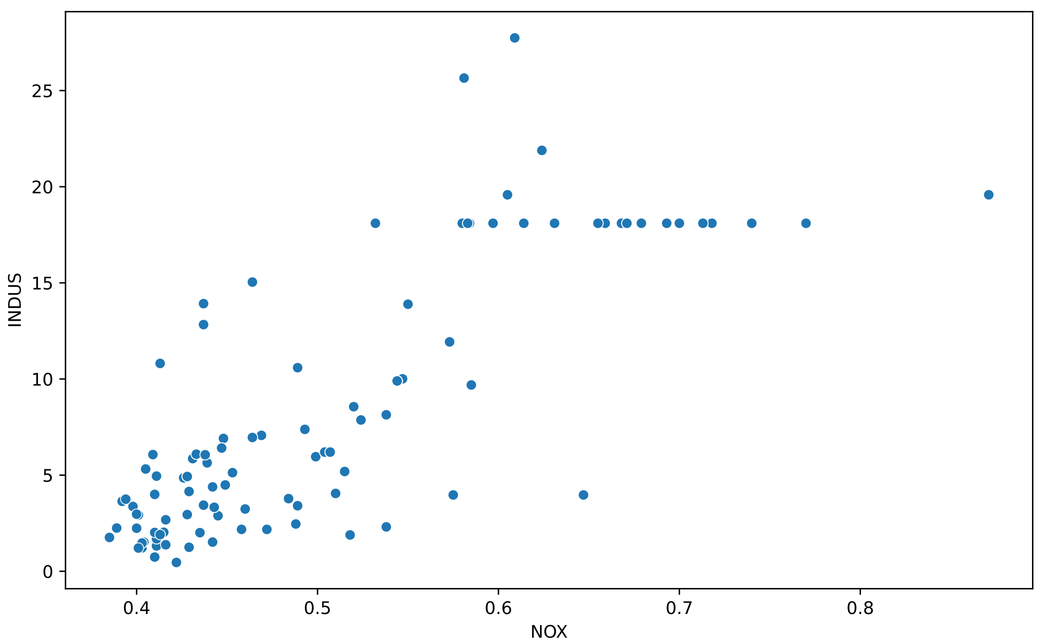
```
## Group_AGE     low  medium  high
## CHAS
## bounds river    3       8    24
## otherwise      88     111   272
```

```
scipy.stats.chi2_contingency(cross_table,correction=True)
```

```
## (2.5116464414864033, 0.2848412644431747, 2, array([[  6.2944664,   8.2312253,  20.4743083],
##        [ 84.7055336, 110.7687747, 275.5256917]]))
```

there is no relationship between Nitric oxide concentrations and proportion of non-retail business acres per town?

```
sns.scatterplot(x='NOX',y='INDUS',data=df)
plt.show()
```



H_0: there are no relationship between Nitric oxide concentrations and proportion of non-retail business acres per town H_1: there are relationship between Nitric oxide concentrations and proportion of non-retail business acres per town Use alpha = 0.05

```
scipy.stats.pearsonr(df['NOX'],df['INDUS'])
```

```
## (0.7636514469209157, 7.913361061233745e-98)
```

Conclusion:

Since the p-value < 0.05, we reject the Null hypothesis and conclude that there are a relationship between Nitric oxide concentrations and proportion of non-retail business acres per town.

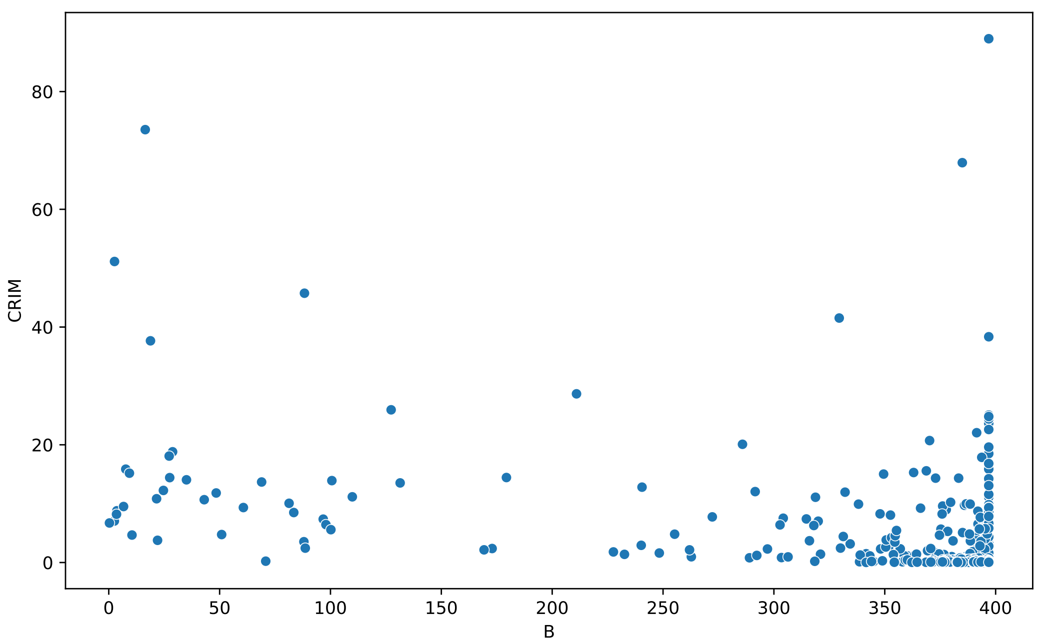
there is no relationship between the proportion of blacks by town and crime rate by town?

H_0: there are no relationship between the proportion of blacks by town and crime rate by town.

H_1: there are relationship between the proportion of blacks by town and crime rate by town.

Use alpha = 0.05

```
sns.scatterplot(x='B',y='CRIM',data=df)
plt.show()
```




```
scipy.stats.pearsonr(df['B'],df['CRIM'])
```

```
## (-0.3850639419942238, 2.4872739737731073e-19)
```

Conclusion:

Since the p-value < 0.05, we reject the Null hypothesis and conclude that there are a relationship between the proportion of blacks by town and crime rate by town.