[Assigning Null (H0) and Alternate Hypothesis (H1) for A and B ingredients]

1.1 Null Hypothesis and Alternate Hypothesis for A and B Ingredients

Null Hypothesis and Alternate Hypothesis for A and B Ingredients

H0- There is no impact on ReliefHours due to ingredient A ($All the population means are equal$ ie u1=u2=u3)

H1- There is a significant impact on ReliefHours due to ingredient A ($At least one of the population means are unequal.$)

H0- There is no impact on ReliefHours due to ingredient B ($All the population means are equal$ ie u1=u2=u3)

H1- There is a significant impact on ReliefHours due to ingredient B ($At least one of the population means are unequal.$)

1.2 One way Anova for A Variable

```
from scipy.stats import shapiro,levene,mannwhitneyu,wilcoxon
from statsmodels.formula.api import ols
from statsmodels.stats.anova import anova_lm
from statsmodels.stats.multicomp import MultiComparison
```

```
df=pd.read_csv('Fever.csv')
df.head()
```

	A	B	Volunteer	Relief
0	1	1	1	2.4
1	1	1	2	2.7
2	1	1	3	2.3
3	1	1	4	2.5

Loading the data and importing the required libraries

```
df.shape
```

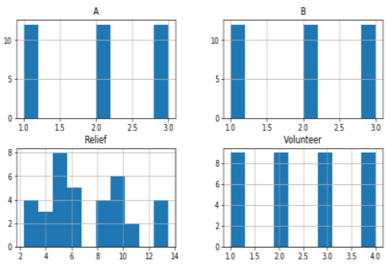
```
(36, 4)
```

```
df['B']= pd.Categorical(df['B'])
df['A']= pd.Categorical(df['B'])
```

```
df.describe(include="all").T
```

	count	mean	std	min	25%	50%	75%	max
A	36.0	2.000000	0.828079	1.0	1.000	2.0	3.000	3.0
B	36.0	2.000000	0.828079	1.0	1.000	2.0	3.000	3.0
Volunteer	36.0	2.500000	1.133893	1.0	1.750	2.5	3.250	4.0
Relief	36.0	7.183333	3.272090	2.3	4.675	6.0	9.325	13.5

```
from pylab import rcParams
rcParams['figure.figsize'] = 10,5
df.hist();
```




```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36 entries, 0 to 35
Data columns (total 4 columns):
A            36 non-null int64
B            36 non-null int64
Volunteer    36 non-null int64
Relief       36 non-null float64
dtypes: float64(1), int64(3)
memory usage: 1.2 KB
```

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

```
3    12
2    12
1    12
Name: A, dtype: int64
```

Performing Some EDA on the Data

```
shapiro(a1)
```

```
(0.7686296701431274, 0.004211828112602234)
```

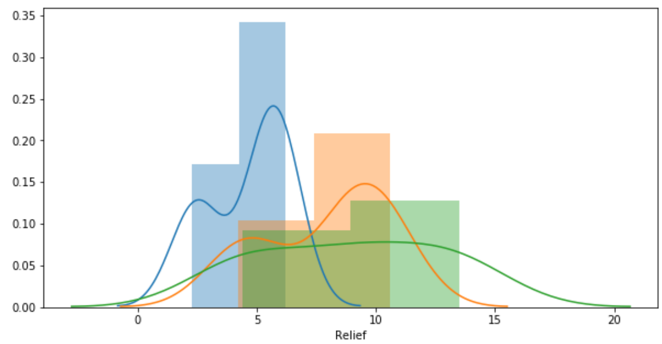
```
shapiro(a2)
```

```
(0.728706955909729, 0.001616060733795166)
```

```
shapiro(a3)
```

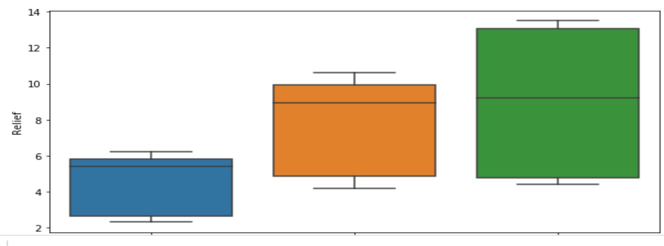
```
(0.847996175289154, 0.03468279168009758)
```

```
sns.distplot(df.loc[df['A'] == 1]['Relief'])
sns.distplot(df.loc[df['A'] == 2]['Relief'])
sns.distplot(df.loc[df['A'] == 3]['Relief'])
plt.show()
```



Performing Shapiro test, so from the above Shapiro test we can find the data is not normal as Shapiro test reject null hypothesis H0

```
sns.boxplot(df['A'],df['Relief'])
plt.show()
```




```
levene(a1.Relief,a2.Relief,a3.Relief)
```

```
LeveneResult(statistic=4.511350350740447, pvalue=0.018535088623493387)
```

As the P-value for both Shapiro and Levene test are below 0.05, In this case we reject Null Hypothesis and we accept Alternate Hypothesis, so the dataset is not normally distributed and the variance is not equal.

As Both Shapiro and Levene test reject H0 we need to do Non –Parametric Test.

```
unique = df['A'].unique()
unique
```

```
array([1, 2, 3], dtype=int64)
```

```
kruskalwallis(np.array(df['Relief'][df['A'] == unique[0]]),
              np.array(df['Relief'][df['A'] == unique[1]]),
              np.array(df['Relief'][df['A'] == unique[2]]))
```

```
KruskalResult(statistic=25.645036693704128, pvalue=2.6992992738200464e-06)
```

We are performing the non-parametric test (kriskalwallis test) .From the Kriskalwallis test we can find that the P value is less than 0.05, so we are rejecting H0, so the population means are not equal.

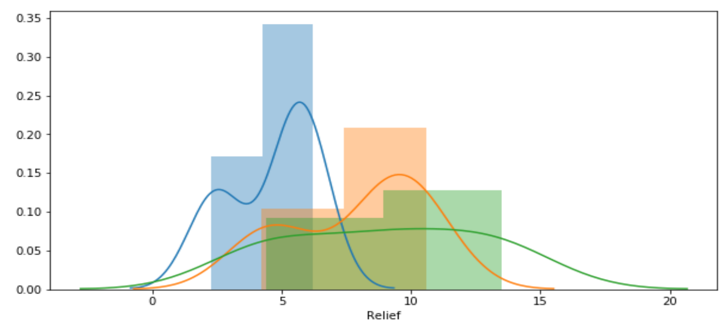
```
mc = MultiComparison(df['Relief'], df['A'])
result = mc.tukeyhsd()

print(result)
print(mc.groupsunique)
```

```
Multiple Comparison of Means - Tukey HSD, FWER=0.05
===================================================
group1 group2 meandiff p-adj   lower  upper  reject
---------------------------------------------------
     1      2      3.3 0.0164  0.5374 6.0626   True
     1      3     4.35 0.0014  1.5874 7.1126   True
     2      3     1.05 0.6164 -1.7126 3.8126  False
---------------------------------------------------
[1 2 3]
```

We are performing multicomparison test as we reject the H0 in the non-parametric test, here with multicomparison test we can find the comparison of population of each groups.

1.3 One way Anova for B Variable

```
sns.distplot(df.loc[df['B'] == 1]['Relief'])
sns.distplot(df.loc[df['B'] == 2]['Relief'])
sns.distplot(df.loc[df['B'] == 3]['Relief'])
plt.show()
```

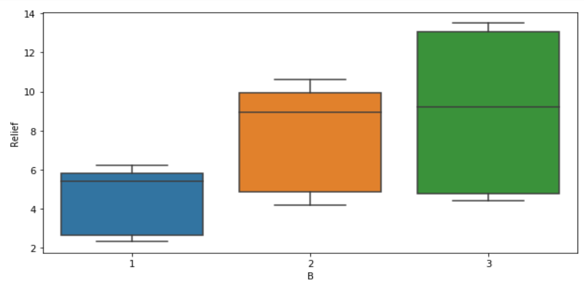



```
for name in df['B'].unique():
    print (shapiro(df['Relief'][df['B'] == name]))
```

```
(0.7656338810920715, 0.003910908009856939)
(0.8065383434295654, 0.01112875435501337)
(0.8446834087371826, 0.031581711024045944)
```

Performing Shapiro test, so from the above Shapiro test we can find the data is not normal as Shapiro test reject null hypothesis H0

```
sns.boxplot(df['B'],df['Relief'])
plt.show()
```




```
levene(b1.Relief,b2.Relief,b3.Relief)
```

```
LeveneResult(statistic=2.941356517208818, pvalue=0.06675699295483081)
```

As the P-value for levene test is above 0.05, In this case we failed to reject Null Hypothesis

As Both Shapiro and Levene test reject H0 we need to do Non –Parametric Test.

```
unique = df['B'].unique()
unique
```

```
[1, 2, 3]
Categories (3, int64): [1, 2, 3]
```

```
kruskalwallis(np.array(df['Relief'][df['B'] == unique[0]]),
              np.array(df['Relief'][df['B'] == unique[1]]),
              np.array(df['Relief'][df['B'] == unique[2]]))
```

```
KruskalResult(statistic=7.3755954680056695, pvalue=0.025027057634532866)
```

We are performing the non-parametric test (kriskalwallis test) .From the Kriskalwallis test we can find that the P value is less than 0.05, so we are rejecting H0, so the population means are not equal.

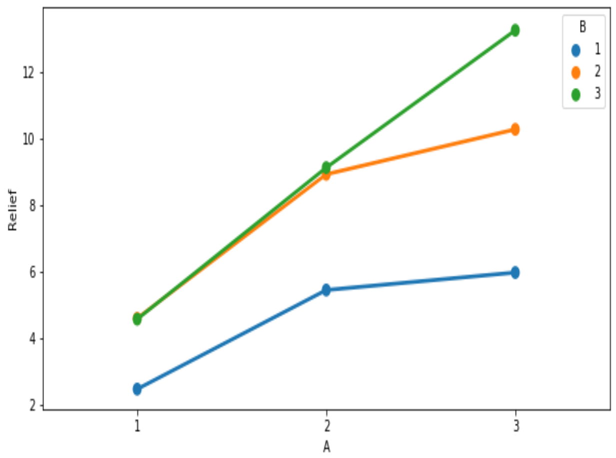
```
mc = MultiComparison(df['Relief'], df['B'])
result = mc.tukeyhsd()

print(result)
print(mc.groupsunique)
```

```
Multiple Comparison of Means - Tukey HSD, FWER=0.05
====================================================
group1 group2 meandiff p-adj   lower  upper  reject
----------------------------------------------------
     1      2      3.3 0.0164  0.5374 6.0626   True
     1      3     4.35 0.0014  1.5874 7.1126   True
     2      3     1.05 0.6164 -1.7126 3.8126  False
----------------------------------------------------
[1 2 3]
```

We are performing multicomparison test due to we reject the H0 in the non-parametric test, here with multicomparison test we can find the comparison of population of each groups.

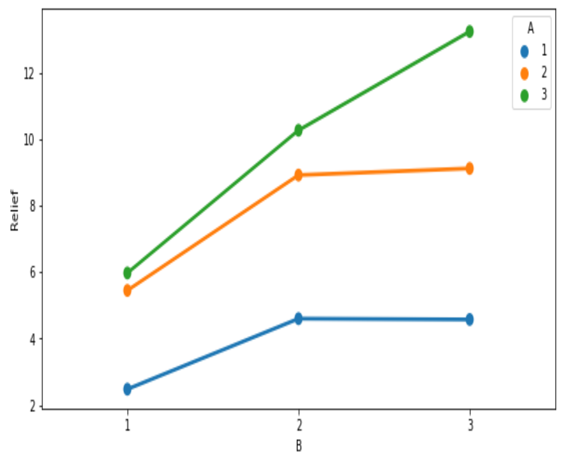
1.4 Analyse the effects of one variable on another with the help of an interaction plot. What is the interaction between the two treatments?

```
sns.pointplot(x = "A",y = "Relief",data = df,hue='B',ci=None)

plt.show() ## from the below result we can come to a conclusion that there is c
```




```
sns.pointplot(x = "B",y = "Relief",data = df,hue='A',ci=None)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x24740648>
```




```
#Interaction Test
formula = 'Relief ~ C(A)+C(B)+C(A):C(B)'
model = ols(formula, df).fit()
aov_table = anova_lm(model)
print(aov_table)
```

	df	sum_sq	mean_sq	F	PR(>F)
C(A)	2.0	220.020	110.010000	1827.858462	1.514043e-29
C(B)	2.0	123.660	61.830000	1027.329231	3.348751e-26
C(A):C(B)	4.0	29.425	7.356250	122.226923	6.972083e-17
Residual	27.0	1.625	0.060185	NaN	NaN

Performed Interaction test graphically and statistically.

There is interaction between ingredients A and B.

1.5 Perform a two-way ANOVA based on the different ingredients (variable ‘A’ & ‘B’) with the variable 'Relief' and state your results.

```
formula = 'Relief ~ C(A)+C(B)'
model = ols(formula, df).fit()
aov_table = anova_lm(model)
print(aov_table)
```

	df	sum_sq	mean_sq	F	PR(>F)
C(A)	2.0	220.02	110.010000	109.832850	8.514029e-15
C(B)	2.0	123.66	61.830000	61.730435	1.546749e-11
Residual	31.0	31.05	1.001613	NaN	NaN

Performed Two way anova as the P value is less than 0.05 we reject Ho

1.6 Mention the business implications of performing ANOVA for this particular case study.

1.When 3rd level of ingredient A is used with 3rd level of ingredient B the relief time is high.

2. When 1st level of ingredient A is used with 1st level of ingredient B the relief time is lesser.

3.Among the three levels of ingredient A level 2 and 3 approximately have same cure time.

4.Among the three levels of ingredient B level 2 and 3 approximately have same cure time.

5.The relief time drastically increases when A with level 1 or 2 or 3 is used with level 3 of B or vice versa.

2.1 Perform exploratory data analysis on the dataset. Showcase some charts, graphs.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from statsmodels.tools.eval_measures import rmse
from statsmodels.multivariate.pca import PCA
import warnings
warnings.filterwarnings("ignore")
from scipy import stats
```

First we are importing the required Libraries for our project.

```
data= pd.read_csv('Income.csv')
data.head()
```

	WorkingHoursWife	WifeAge	EducationWife	WifeHourEarnings	WifeWage	WorkingHoursHusband	HusbandAge	EducationHusband	HusbandWage	Educatio
0	1610	32	12	3.3540	2.65	2708	34	12	4.0288	
1	1656	30	12	1.3889	2.65	2310	30	9	8.4416	
2	1980	35	12	4.5455	4.04	3072	40	12	3.5807	
3	456	34	12	1.0965	3.25	1920	53	10	3.5417	
4	1568	31	14	4.5918	3.60	2000	32	12	10.0000	

Here we are reading the data we are going to do the EDA.

```
data.shape
```

```
(753, 14)
```

It shows the data has 14 columns and 753 data points

```
data.describe(include='all').T
```

	count	mean	std	min	25%	50%	75%	max
WorkingHoursWife	753.0	740.576361	871.314216	0.0000	0.0000	288.0000	1516.0000	4950.000
WifeAge	753.0	42.537849	8.072574	30.0000	36.0000	43.0000	49.0000	60.000
EducationWife	753.0	12.286853	2.280246	5.0000	12.0000	12.0000	13.0000	17.000
WifeHourEarnings	753.0	2.374565	3.241829	0.0000	0.0000	1.6250	3.7879	25.000
WifeWage	753.0	1.849734	2.419887	0.0000	0.0000	0.0000	3.5800	9.980
WorkingHoursHusband	753.0	2267.270916	595.566649	175.0000	1928.0000	2164.0000	2553.0000	5010.000
HusbandAge	753.0	45.120850	8.058793	30.0000	38.0000	46.0000	52.0000	60.000
EducationHusband	753.0	12.491368	3.020804	3.0000	11.0000	12.0000	15.0000	17.000
HusbandWage	753.0	7.482179	4.230559	0.4121	4.7883	6.9758	9.1667	40.509
EducationWifeMother	753.0	9.250996	3.367468	0.0000	7.0000	10.0000	12.0000	17.000
EducationWifeFather	753.0	8.808765	3.572290	0.0000	7.0000	7.0000	12.0000	17.000

From the above describe function we can find the count of the variables as 753 and we can easily come to know about the outliers in many of the features. And also there is no any categorical variables present.

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 753 entries, 0 to 752
Data columns (total 14 columns):
WorkingHoursWife       753 non-null int64
WifeAge                753 non-null int64
EducationWife          753 non-null int64
WifeHourEarnings       753 non-null float64
WifeWage               753 non-null float64
WorkingHoursHusband    753 non-null int64
HusbandAge             753 non-null int64
EducationHusband       753 non-null int64
HusbandWage            753 non-null float64
EducationWifeMother    753 non-null int64
EducationWifeFather    753 non-null int64
UnemploymentRate       753 non-null float64
WifeExperience         753 non-null int64
```

This shows the data types of each variable

```
data.isnull().sum()
```

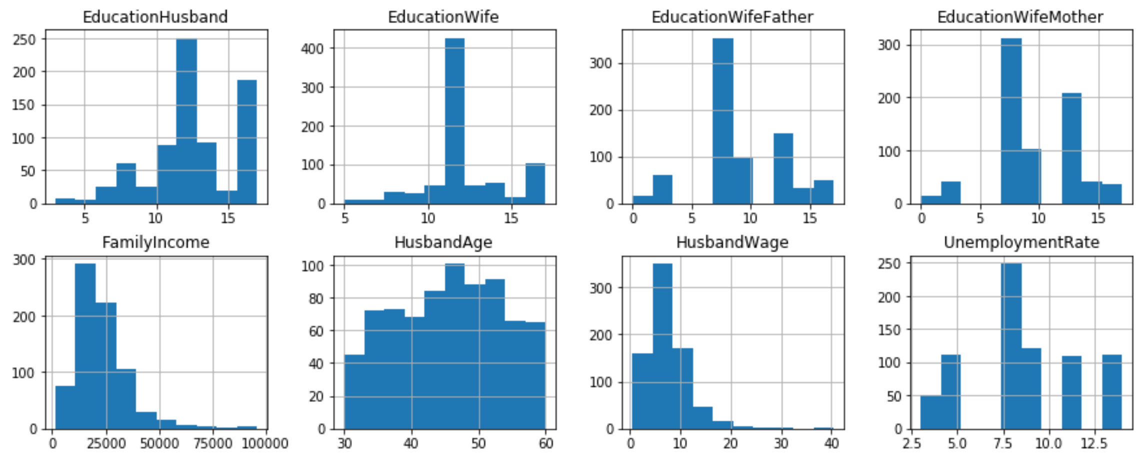
```
WorkingHoursWife         0
WifeAge                  0
EducationWife            0
WifeHourEarnings         0
WifeWage                 0
WorkingHoursHusband      0
HusbandAge               0
EducationHusband         0
HusbandWage              0
EducationWifeMother      0
EducationWifeFather      0
UnemploymentRate         0
WifeExperience           0
FamilyIncome             0
dtype: int64
```

This shows there is no null values in the dataset

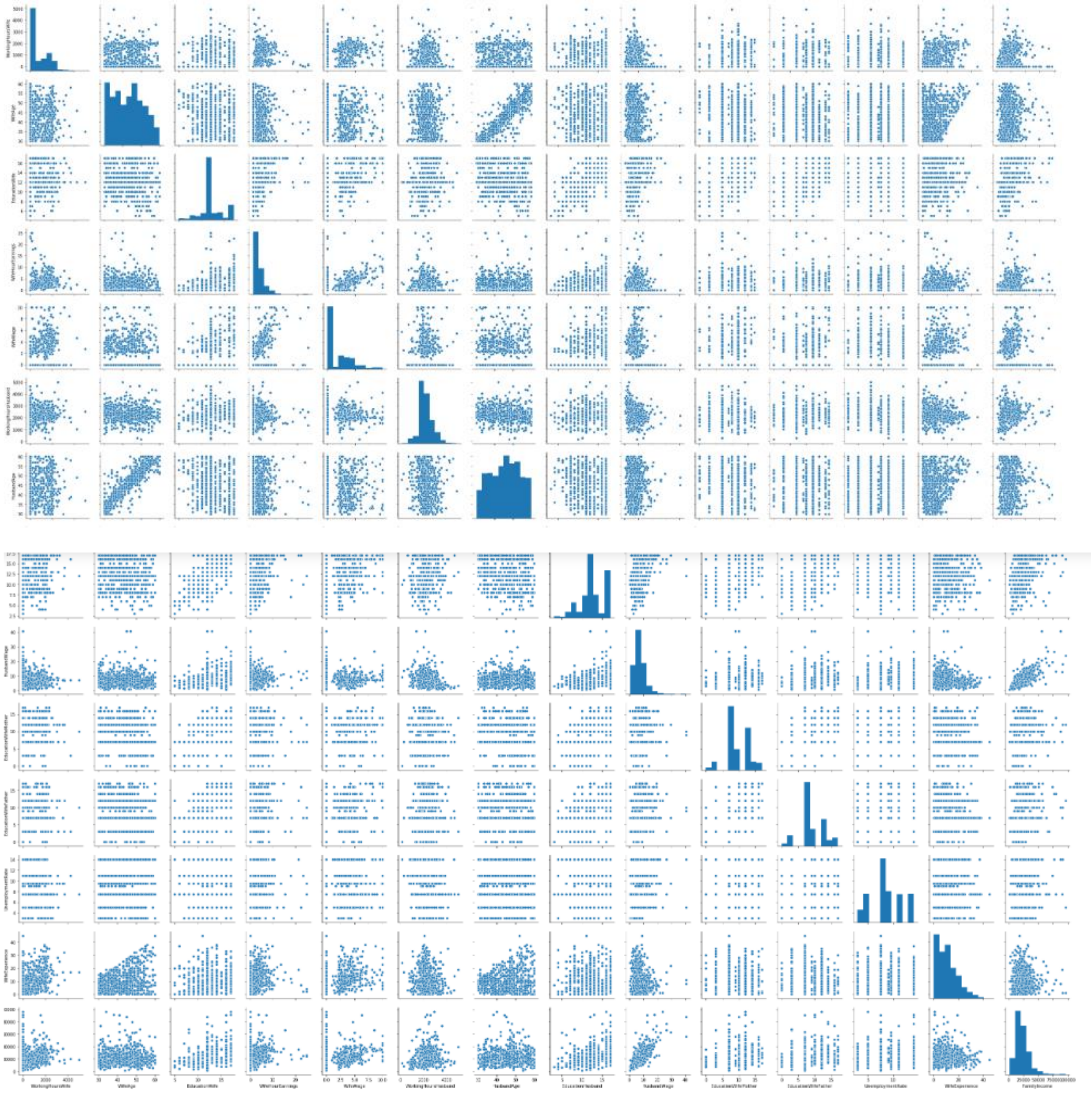
```
dups = data.duplicated()
dups.value_counts()
```

```
False    753
dtype: int64
```

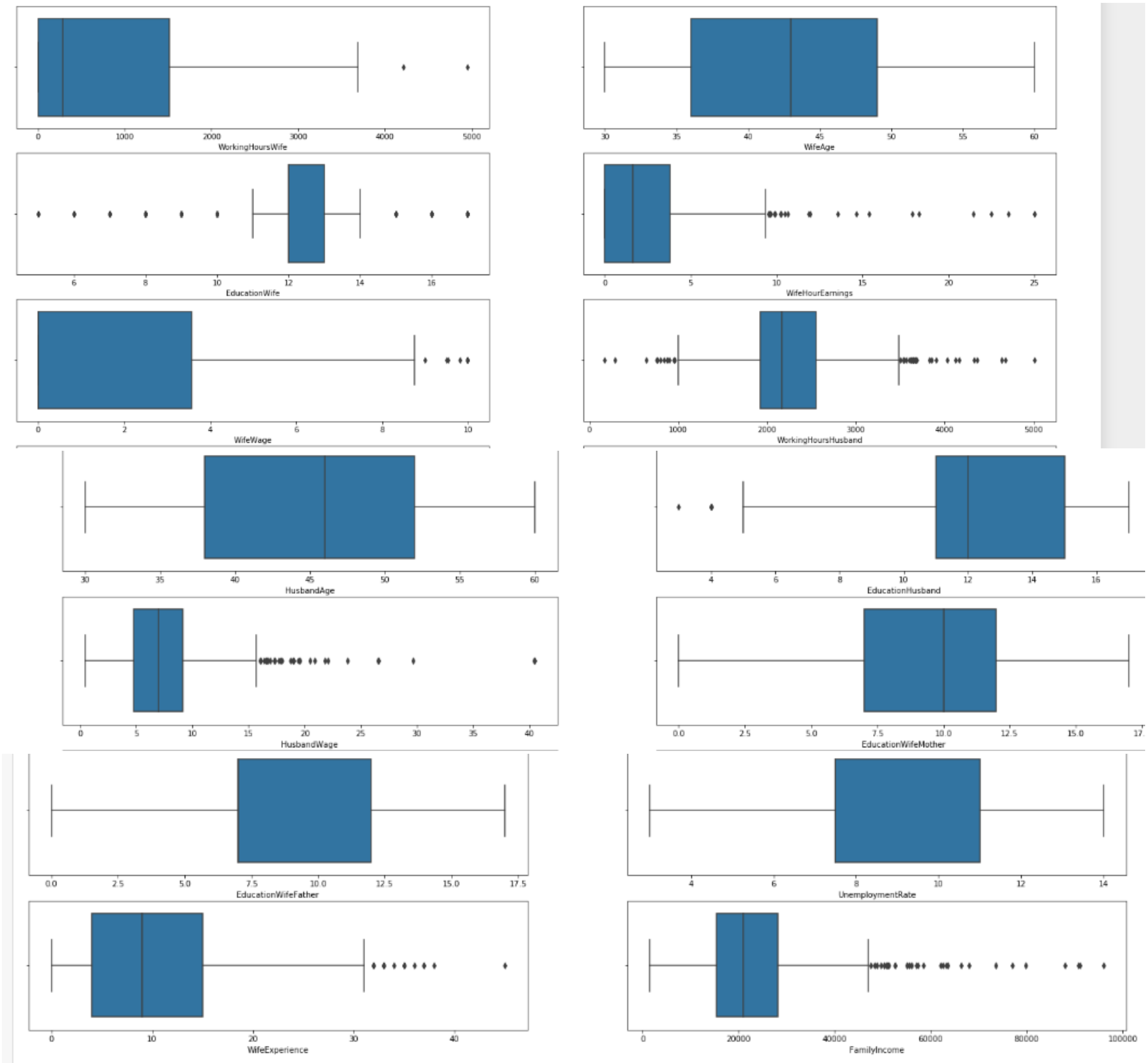
This shows there is no duplicate values in the dataset



This shows the distribution of the each feature in the data, from this we can find some features like educationwife are having normal distribution and some feature like wifehourearnings are right skewed, etc.



The Above Pair plot represents the relationship of each feature with each other in a graphical representation



The above boxplot represents the features having outliers, so we need to handle the outliers

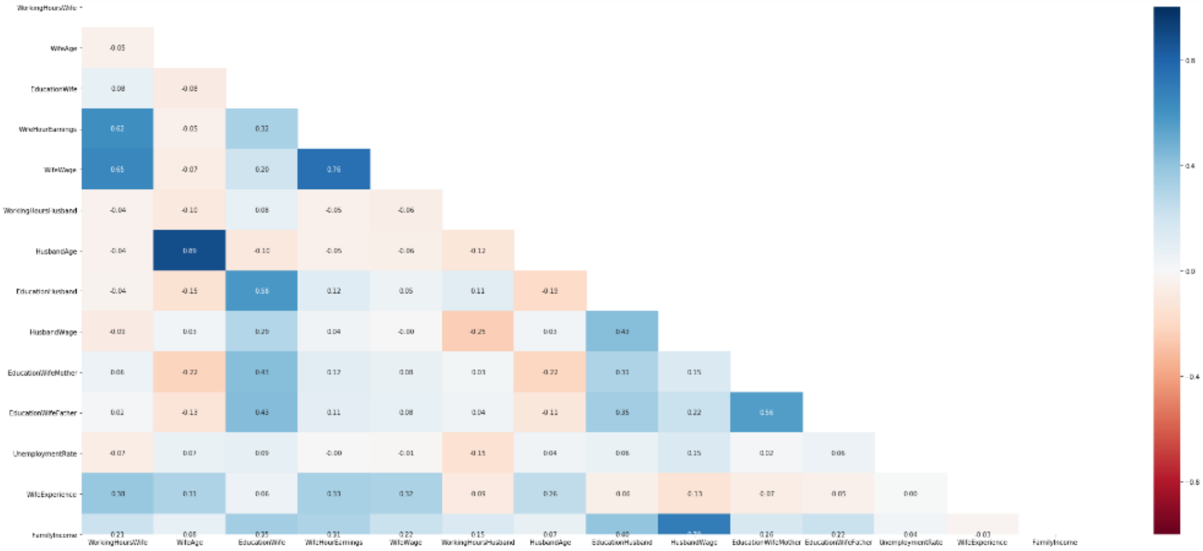
```
z = np.abs(stats.zscore(data))
data1= data[(z <=3).all(axis=1)]
```

```
data1.shape
```

```
(695, 14)
```

After removing the outlier the sample size reduced from 753 to 695

2.2 **Perform** Multicollinearity Test.



The heatmap represents the multicollinearity between the features, so we can find there is a strong relationship between wifeage and wifeworkinghours, Familyincome and husbandwage ,etc and also find there is a high multicollinearity between the independent variables , so this will result in reducing the performance for linear regression model

2.3 Perform Multiple Linear Regression

OLS Regression Results

Dep. Variable:	FamilyIncome	**R-squared:**	0.724
Model:	OLS	**Adj. R-squared:**	0.719
Method:	Least Squares	**F-statistic:**	137.6
Date:	Fri, 19 Feb 2021	**Prob (F-statistic):**	1.55e-180
Time:	11:24:35	**Log-Likelihood:**	-6916.5
No. Observations:	695	**AIC:**	1.386e+04
Df Residuals:	681	**BIC:**	1.392e+04
Df Model:	13		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2.049e+04	2038.049	-10.054	0.000	-2.45e+04	-1.65e+04
WorkingHoursWife	2.5356	0.337	7.516	0.000	1.873	3.198
WifeAge	148.9450	54.883	2.714	0.007	41.184	256.706
EducationWife	114.0232	126.304	0.903	0.367	-133.970	362.016
WifeHourEarnings	712.3784	133.194	5.348	0.000	450.859	973.898
WifeWage	-19.6182	146.834	-0.134	0.894	-307.921	268.684
WorkingHoursHusband	6.4166	0.388	16.541	0.000	5.655	7.178
HusbandAge	33.5834	53.328	0.630	0.529	-71.123	138.290
EducationHusband	48.7249	90.140	0.541	0.589	-128.261	225.710
HusbandWage	2221.1076	70.275	31.606	0.000	2083.125	2359.090
EducationWifeMother	40.2344	74.190	0.542	0.588	-105.434	185.903
EducationWifeFather	19.6868	70.343	0.280	0.780	-118.429	157.803
UnemploymentRate	-66.6494	64.310	-1.036	0.300	-192.919	59.621
WifeExperience	-94.4798	30.644	-3.083	0.002	-154.648	-34.311

Omnibus:	165.961	**Durbin-Watson:**	1.979
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	379.818
Skew:	1.267	**Prob(JB):**	3.34e-83
Kurtosis:	5.587	**Cond. No.**	2.56e+04

Varnings:
1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
2] The condition number is large, 2.56e+04. This might indicate that there are
trong multicollinearity or other numerical problems.

From the above multiple linear regression test we can find that the R2 value is 0.72 , so it is not a bad model to be not considered and also Durbin Watson test parameter are around 1.97 so it clearly explains there is no autocorrelation within the error.

```
ypred = model.predict(X)
print(ypred)
```

```
0      19078.498082
1      24713.569856
2      22732.428115
3       9787.446302
4      29066.018483
           ...
748    28637.204143
749    10079.141497
750     6605.246300
751    28990.090627
752    20638.631714
Length: 695, dtype: float64
```

This shows the Model prediction (Ypred)

```
from sklearn import metrics
print('RMSE:',np.sqrt(metrics.mean_squared_error(Y,ypred)))
```

```
RMSE: 5079.037352292563
```

The linear regression evaluation metrics RMSE score is calculated

2.4 Perform PCA

```
from scipy.stats import zscore #Scaling
data1_scaled=data1.drop(['FamilyIncome'],axis =1).apply(zscore)
data1_scaled.head()
```

	WorkingHoursWife	WifeAge	EducationWife	WifeHourEarnings	WifeWage	WorkingHoursHusband	HusbandAge	EducationHusband	HusbandWage	Educat
0	1.093541	-1.294315	-0.107495	0.524951	0.443267	0.810073	-1.361120	-0.155443	-0.923598	
1	1.149369	-1.546962	-0.107495	-0.282139	0.443267	0.081828	-1.860580	-1.181070	0.392183	
2	1.542597	-0.915345	-0.107495	1.014315	1.082884	1.476106	-0.611929	-0.155443	-1.057210	
3	-0.307027	-1.041668	-0.107495	-0.402232	0.719361	-0.631778	1.011318	-0.839194	-1.068839	
4	1.042567	-1.420638	0.820564	1.033331	0.880415	-0.485397	-1.610850	-0.155443	0.856857	

We are scaling the Independent features before PCA is done

```
tot = sum(eig_vals)
var_exp = [( i /tot ) * 100 for i in sorted(eig_vals, reverse=True)]
cum_var_exp = np.cumsum(var_exp)
print("Cumulative Variance Explained", cum_var_exp)
```

```
Cumulative Variance Explained [ 22.73892828  41.99766504  56.4499376   65.79842313  72.93370457
  79.75441663  84.87649484  88.7157527   92.02246394  94.88239432
  97.40019728  99.16434883 100.        ]
```

Here we are finding the cumulative Eigen values to decide on the number of components for reducing the dimensionality of the features with more importance and reducing the number of features.

```
pc = PCA(data1.drop(['FamilyIncome'],axis =1),ncomp=5)
```

```
pc.loadings.T
```

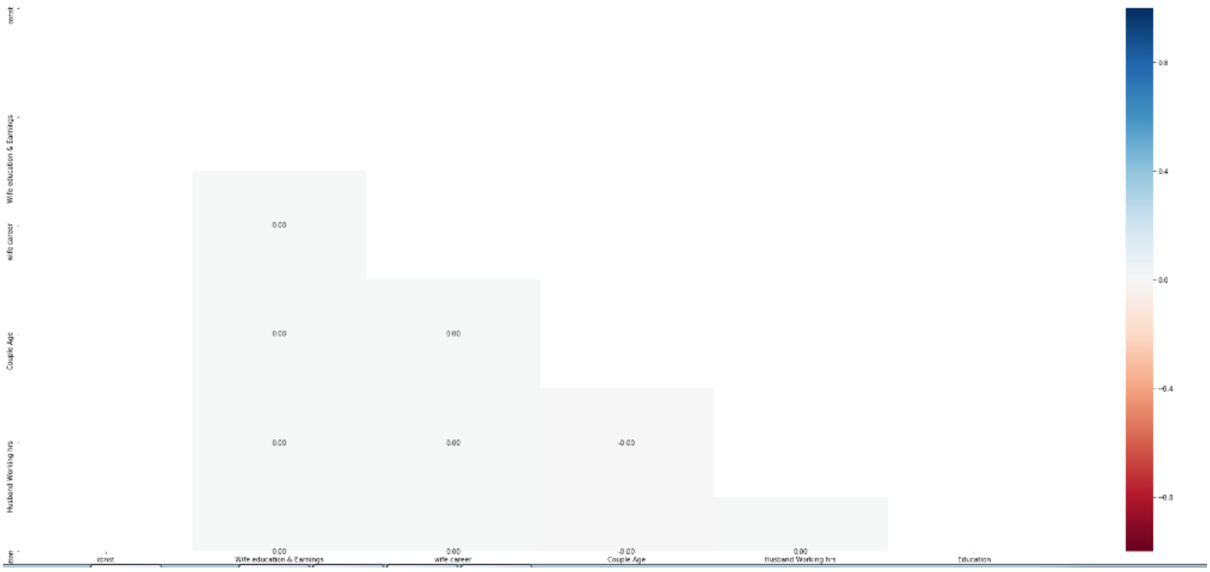
	WorkingHoursWife	WifeAge	EducationWife	WifeHourEarnings	WifeWage	WorkingHoursHusband	HusbandAge	EducationHusband	HusbandWage
comp_0	-0.274770	0.222394	-0.414237	-0.376528	-0.343131	-0.016104	0.227446	-0.342966	-0.185147
comp_1	0.417339	0.289365	-0.104792	0.373690	0.395290	-0.110839	0.288204	-0.229956	-0.157980
comp_2	0.188538	-0.519051	-0.242074	0.079589	0.136349	0.197802	-0.506635	-0.230903	-0.385772
comp_3	-0.034512	0.224515	0.172303	-0.062303	-0.095245	0.665030	0.209644	0.076242	-0.375374
comp_4	0.035592	0.000072	-0.164741	-0.106641	-0.058789	-0.298247	0.035592	-0.456048	-0.344651

Here we are loading the components to the data

```
pc.factors.rename(columns = {'comp_0':'Wife education & Earnings', 'comp_1':'wife career',
                             'comp_2':'Couple Age', 'comp_3':'Husband Working hrs','comp_4':'Education'}, inplace = True)
pc.factors.head()
```

	const	Wife education & Earnings	wife career	Couple Age	Husband Working hrs	Education
0	1.0	-0.024398	0.008031	0.068941	0.026371	0.003703
1	1.0	-0.007002	-0.007226	0.061349	-0.057136	-0.005086
2	1.0	-0.030316	0.030847	0.063491	0.048648	-0.003724
3	1.0	0.022343	0.011443	0.028643	0.001062	0.004640
4	1.0	-0.063105	-0.011867	0.027046	-0.038278	0.029523

Here we are renaming the Components wit highest variance captured. Now the number of features has been reduced to 5 from 14 features



The Multicollinearity result after the PCA has been done. Now we can see clearly there is no correlation between the independent variables.

2.5 Perform Multiple Linear egression with “FamilyIncome” as the dependent variable and the PCA extracted components as the Independent variables.

OLS Regression Results

Dep. Variable:	FamilyIncome	R-squared:	0.455
Model:	OLS	Adj. R-squared:	0.451
Method:	Least Squares	F-statistic:	115.0
Date:	Fri, 19 Feb 2021	Prob (F-statistic):	2.33e-88
Time:	22:02:42	Log-Likelihood:	-7153.3
No. Observations:	695	AIC:	1.432e+04
Df Residuals:	689	BIC:	1.435e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	2.192e+04	272.057	80.569	0.000	2.14e+04	2.25e+04
Wife education & Earnings	-1.127e+05	7172.199	-15.707	0.000	-1.27e+05	-9.86e+04
wife career	-1694.9686	7172.199	-0.236	0.813	-1.58e+04	1.24e+04
Couple Age	-8.638e+04	7172.199	-12.044	0.000	-1e+05	-7.23e+04
Husband Working hrs	-7103.9073	7172.199	-0.990	0.322	-2.12e+04	6978.081
Education	-9.685e+04	7172.199	-13.504	0.000	-1.11e+05	-8.28e+04

Omnibus:	114.968	Durbin-Watson:	2.041
Prob(Omnibus):	0.000	Jarque-Bera (JB):	236.869
Skew:	0.933	Prob(JB):	3.67e-52
Kurtosis:	5.168	Cond. No.	26.4

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

2.6 Comment on the Model thus built using the Principal Components and with 'FamilyIncome'.

The Above Multiple linear regression is done after PCA. Now we are getting the R2 value as 0.455, this represents the model is very bad as the independent variable not able to explain the variance of the dependent variable effectively. Actually the data need more sample for the good model prediction

```
from sklearn import metrics
print('RMSE:',np.sqrt(metrics.mean_squared_error(Y,ypred_pca)))
```

```
RMSE: 7141.172598816365
```

We can find from rmse value also it is increased from earlier rmse value before PCA is done, so it is not a good model.

2.7 Mention the business implication and interpretation of the models.

1. The Husband wage is very significant i.e. 1 unit increase in his wage increases the family income the most i.e. by plus 2221.

2. With every 1 year increase in wife experience the family income reduces by 94 euros.

3.With increase in unemployment rate the family income reduces by 66 euros.

4.working hours of wife has nearly no impact on family income.

5.With 1yr increase in wife age the family income increases by 148 euros.

6.With increase in wife education the family income increases by 114 euros.

7. If wife hourly income increases by 1 unit the family income increases by 712 euros.

8. Wife wage, Husband age, Education husband, Educationwifemother and Educationwifefather is not significant in family income.

9. With increase in working hours of husband by 1hr the family income increases by 6euro.