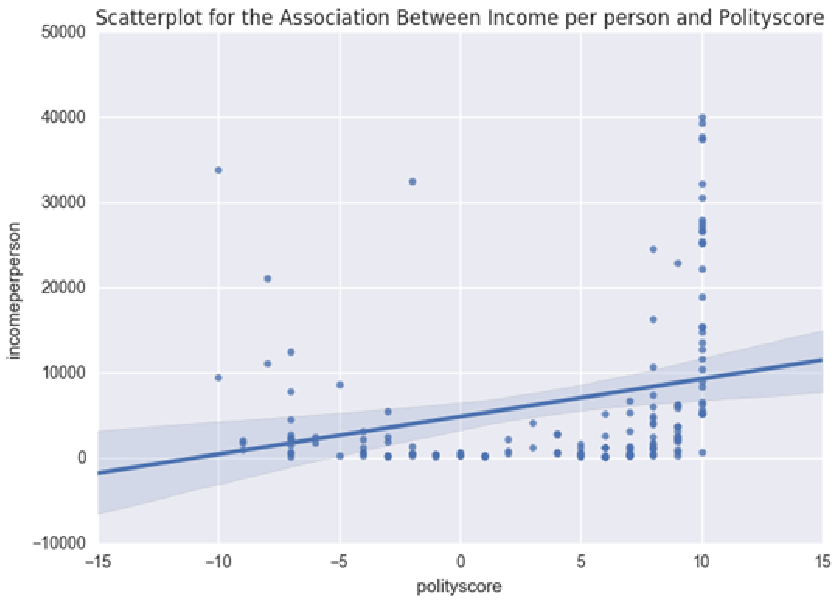
Data Analysis with Python

Week 3 Correlation

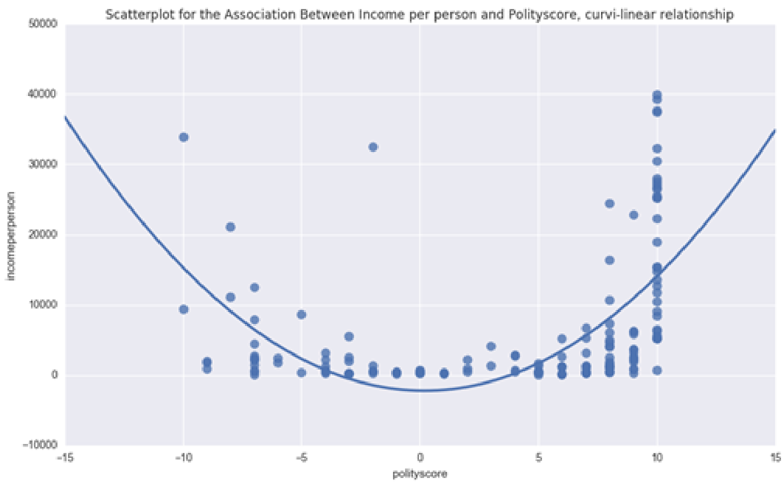
Introduction

Pearson's correlation coefficient is calculated for the association between income per person and polity score. The correlation was found to be:

- Correlation (0.37885828974133834)
- The p value was found to be 0.10968995828465959 so the association was not found to be statistically significant.



The scatter plot revealed a curvilinear relationship meaning that Pearson's correlation is not appropriate as it pertains to linear relationships, not curvilinear relationships.



A curvi-linear relationship is clearly visible.

Conclusion

A curvilinear relationship is seen between Income per person and correlation. Pearsons correlation coefficient (as it measures correlation of linear associations) is not a suitable method to judge the association of income per person and polity score.

Code

```
import scipy

import seaborn

import matplotlib.pyplot as plt

data.columns.values

##'incomeperperson''polityscore'

##pre-process the data

data['incomeperperson']=data['incomeperperson'].replace(' ', numpy.nan)

data['polityscore']=data['polityscore'].replace(' ', numpy.nan)

##

scat1 = seaborn.regplot(x="polityscore", y="incomeperperson", fit_reg=True, data=data)

plt.xlabel('polityscore')

plt.ylabel('incomeperperson')

plt.title('Scatterplot for the Association Between Income per person and Polityscore')

data_test=data.dropna()

print ('association between income per person and polity score and internetuserate')

print (scipy.stats.pearsonr(data_test['incomeperperson'], data_test['polityscore']))

##print (scipy.stats.pearsonr(data_test['incomeperperson'], data_test['polityscore']))

##(0.37885828974133834, 0.10968995828465959)

##From the scatterplot there appeats to a curvo linear relationship,

##the correlation is useless or assessing non linear relationships

seaborn.lmplot(x="polityscore", y="incomeperperson", data=data,

    order=2, ci=None, scatter_kws={"s": 80});

plt.xlabel('polityscore')

plt.ylabel('incomeperperson')

plt.title('Scatterplot for the Association Between Income per person and Polityscore, curvi-linear relationship')
```

NT8Pwx�S�A�

Feb 28th, 2016

MORE YOU MIGHT LIKE

Logistic regression models week 4

Introduction

The hypothesis we wish to test is that armed forces rate is to investigate whether armed forces rate has an

Week 2 Chi-squared tests

A chi squared test was used to determine if NATO and EU membership (or non membership) had an effect on a country being a democracy (democracy) or a non-democracy (anocracy or autocracy). All code is available here:

Week 1 Anova testing

Week 1 ANOVA testing

This week we run a simple ANOVA test on European countries to see how income per person is affected my NATO and EU membership. All code is available here:

Regress Models

Overview

This weeks upda specification proc multi value model following steps:

effect on whether a country is a democracy or not (democracy_cat_int).

To do this we create a binary response variable by applying the two functions listed in appendix one to first create a categorical variable with three levels and then by applying polityscore_cat_int

to create a binary response variable (1 if a country is a democracy, o if it is not).

Polity scores have the following breakdowns:

· Polity score range 6 to10: Democracy

· Polity score range -5 to 5: Anocracy

· Polity score range -10 to -6: Autocracy

After our initial investigation we build a second model including the explanatory variables:

· Female employee rate

· Armed forces rate

· Internet user rate

· Urban rate

· Income per person

The aim of this is twofold:

· To see if the additional variables effect on whether a country is a democracy or not.

· To check for evidence of confounding.

A full script to replicate the code can be downloaded here:

(https://github.com/brennap3/Gapminder/blob/master/Gapminder_Analysis_2015.py)

Pre-processing the data

Besides converting the response variable to a binary response variable we also centred all our explanatory variables with their means at zero.

Interpreting our model: Model 1

A summary is obtained of the logistic model (explanatory: Democracy (1, Non Democracy 0) and response variable centred armed forces rate). From the summary the odds ratio and the confidence intervals can be obtained.

For armed forces rate, the values are:

5% 95% OR

https://github.com/brennap3/Gapminder/blob/master/Gapminder_Analysis_2015.py

After calculating the different categories and sub-setting for European countries. A chi squared test is run. The values returned by the test are (appendix 1 shows the code used to run this test):

· Chi-squared statistic13.636363636363637

· P value 0.0034443294821406593

· Our alpha value is 0.05 and are p value is 0.0034443294821406593, this is below the critical value so we reject the null hypothesis (that the relative proportions of one variable are independent of the second variable) that there is no difference between groups. There is a statistically difference between groups (that the relative proportions of one variable are associated with the second variable).

The difference between the groups is shown below

NATO_EU_MEMBERSHIP Nato_And_EU Nato_Not_In_EU Not_In_Nato_In_EU \

polityscore_cat_democracy

Democracy 19 3 5

Not Democracy 1 1 1

NATO_EU_MEMBERSHIP Not_In_Nato_Not_In_EU

polityscore_cat_democracy

Democracy 6

Not Democracy 9

Next we run a series of post hoc tests (these are shown in appendix 2), the corrected p value which we reject the null hypothesis is equal to 0.05/3 or 0.016666 (as we are conducting 3 tests). The 3 different groups we are comparing are:

· "Nato_And_EU", "Nato_Not_In_EU" (p value: 0.94643404330274994)

· "Nato_And_EU" "Not_In_Nato_Not_In_EU" (p value: 0.0014407311825336937)

· "Not_In_Nato_In_EU" "Not_In_Nato_Not_In_EU"(p value: 0.18931317392613722)

https://github.com/brennap3/Gapminder/blob/master/Gapminder_Analysis_2015.py.

The test determines whether there is a difference or not between the means of the different groups or whether the difference is owing to some random variation. The test involves the following steps:

· Compute the group means and standard deviation

· Calculate the within group variance

· Calculate the between group variance

· Compute the F statistic by utilizing the within and between group variance

· Check the significance of the F-statistic

The different groups means, variances and standard deviations tested are:

NATO_EU_MEMBERSHIP

Nato_And_EU 16145.634068

Nato_Not_In_EU 20295.315233

Not_In_Nato_In_EU 23345.322253

Not_In_Nato_Not_In_EU 19767.767749

The different groups variances are tested are:

print (varianceincomeperpersn)

incomeperperson

NATO_EU_MEMBERSHIP

Nato_And_EU 1.586113e+08

Nato_Not_In_EU 3.782496e+08

Not_In_Nato_In_EU 6.775285e+07

Not_In_Nato_Not_In_EU 1.058797e+09

The different groups standard deviations tested are:

incomeperperson

NATO_EU_MEMBERSHIP

Nato_And_EU 12594.097210

Nato_Not_In_EU 19448.639876

Not_In_Nato_In_EU 8231.212114

(A) A multidime hopefully taking ir important explana were apparent in model developed

(B) A qqplot eva that residuals are

© A standardis will be used to sh error

(D) Leverage pl means of quantify independent varia average. Points v can have a profou estimates of regr

The git respositor

https://github.com/blob/master/Gap 5.py

Data Selection a

Data is selected v missing values (N of interest, these

a. 'incomeper variable

b. 'polityscore variable

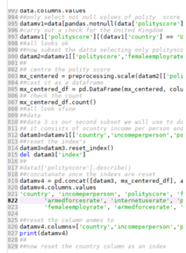
c. 'femaleemp variable

d. 'armedforce variable

e. 'internetuse variable

They are all from dataset. Next we variables. We hop and significance variables ; polity freedom), interne level of economic employee rate (ge armed forces rate militarization) hav person (economic country.

The code to do th



The code is show the full code is av from (

0.559936 0.933543 0.722997

From the summary we can also see that the effect of the variable is statistically significant. So we can say that armed forces rate was significantly associated with democracy such that countries with higher armed forces rate were significantly less likely to be a democracy (OR= 0.722997, 95% CI=0.559936 -0.933543, p=0.013).

Summary

>>> print (lreg1.summary())

Logit Regression Results

==

Dep. Variable: polityscore_cat_int No. Observations: 146

Model: Logit Df Residuals: 144

Method: MLE Df Model: 1

Date: Sun, 14 Feb 2016 Pseudo R-squ.: 0.03797

Time: 19:42:42 Log-Likelihood: -95.756

converged: True LL-Null: -99.536

LLR p-value: 0.005970

===

coef std err z P>|z| [95.0% Conf. Int.]

Intercept 0.2986 0.172 1.735 0.083 -0.039 0.636

armedforcesrate_centred -0.3244 0.130 -2.487 0.013 -0.580 -0.069

===

Odds ratio Model 1:

2.5% 97.5% OR

Intercept 1.313153 3.374140 2.104937

So the only statistically significant difference between groups identified by the post hoc tests is “Nato_And_EU” “Not_In_Nato_Not_In_EU” (p value: 0.0014407311825336937). We can say that the Chi-Square Test of Independence comparing frequencies of one categorical variable (Democracy and Non democracy) for different values of a second categorical variable (NATO EU membership, “Nato_And_EU” “Not_In_Nato_Not_In_EU”), we can say that alternate hypothesis holds true and that the relative proportions of the democracy variable is associated with the NATO EU membership, (“Nato_And_EU” “Not_In_Nato_Not_In_EU”) variable.

Appendix 1

cs1= scipy.stats.chi2_contingency(ct1)

>>> print(cs1)

(13.636363636363637, 0.0034443294821406593, 3L, array([[14.66666667, 2.93333333, 4.4 , 11.],

[5.33333333, 1.06666667, 1.6 , 4.]]))

Appendix2

##Nato_And_EU Not_In_Nato_In_EU

recode3 = {‘Nato_And_EU’:'Nato_And_EU’, 'Not_In_Nato_In_EU’: 'Not_In_Nato_In_EU’}

datasub2['COMP1v3']= datasub2['NATO_EU_MEMBERSHIP']. map(recode3)

ct3=pandas.crosstab(datasub2['polityscore_cat_democracy’], datasub2['COMP1v3'])

print (ct3)

column percentages

colsum=ct3.sum(axis=0)

colpct=ct3/colsum

print(colpct)

##

print ('chi-square value, p value, expected counts’)

cs3= scipy.stats.chi2_contingency(ct3)

print (cs3)

##0.10909090909090913 0.74118150587360399

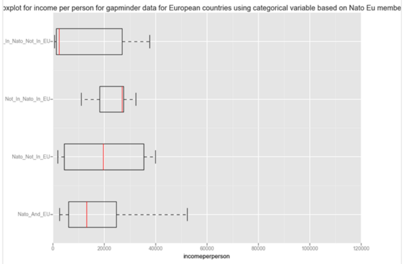
Not_In_Nato_Not_In_EU 32539.161967

Comparing the different groups

Comparing the different groups using multiple boxlpots

Using a boxplot we can see the distribution of the income’s per person for each country in Europe based on the following classes:

1. Nato_And_EU
2. Nato_Not_In_EU
3. Not_In_Nato_In_EU
4. Not_In_Nato_Not_In_EU



The difference in the distribution can be seen from the above plot, witht Not_in_Nato_In EU showing the narrowest variation and the highest median values.

Running the ANOVA analysis

The analysis of variance (and standard deviation) within groups shows that equality of homogeneity assumption maybe violated. Running the ANOVA analysis using the general linear model function we can see that there is evidence obtained that there are differences amongst the different groups. The ANOVA analysis gave an F-statistic of 8.026 and a p value of 4.66e-05 (Prob (F-statistic): 4.66e-05), which is less than our critical value of 0.05. This would indicate that there is a difference in means of the different groups. The model and results are run using the following code:

model1 = smf.ols(formula=‘incomeperperson ~ C(NATO_EU_MEMBERSHIP)’, data=data)

results1 = model1.fit()

print(results1.summary())

##F-statistic: 8.026

##Prob (F-statistic): 4.66e-05

To check the pairwise comparisons, a between the different groups Tukey’s honest significant difference test is run in combination with the ANOVA as a

https://github.com
)

The Summary o

The f tests the nu data can be mode with regression c The alternative hy one of the coeffic non zero. If the f p-value below a t can reject the nul conclude that our useful. As our va statistic): is esse can say our mode will see in a minu particularly usefu

The second value R^2 value. The R^2 of determination a good the model fi in other words ho variance in the da explain. Our mod (R-squared: 0.67 good at fitting the about 67% of the response variable model is quite ver

Diagnostics of t

Using the t-test s hypothesis that th coefficient for a s i.e. it has little influ variable. The alte effect the respons threshold at 0.05, statistic show tha internetuserate_c rate centred) is th significant explan femaleemployrate employee rate ce it has a p-value c than our significa is not statistically userate for every internet use rate, increases by 298 other two variable on the response v and beta values a

t P>|t| [

Intercept 14.351 0.00(7832.993

armedforcesrate 0.559936 0.933543 0.722997

Second Model

In the second model we include all centred explanatory variables in our model, these are:

· Female employee rate

· Armed forces rate

· Internet user rate

· Urban rate

· Income per person

From the summary, we can see that the armed forces rate and internet use rate are statistically significant. Looking at the extracted Odds ratios, we can see that armed forces rate was significantly associated with democracy such that countries with higher armed forces rate were significantly less likely to be a democracy (OR= 0.525738, 95% CI=0.360439-0.766844 , p=0.001). While internet use rate is also significantly associated with democracy such that countries with higher internet usage rates are more likely to be democracies (OR= 1.045871, 95% CI=1.015899 -1.076728 , p=0.003).

Summary second model

```
                    Logit Regression Results
============================================================================================
Dep. Variable:     polityscore_cat_int   No. Observations:        146
Model:                   Logit   Df Residuals:             140
Method:                    MLE   Df Model:                   5
Date:          Sun, 14 Feb 2016   Pseudo R-squ.:          0.2189
Time:                  19:46:12   Log-Likelihood:         -77.750
converged:                  True   LL-Null:               -99.536
                                   LLR p-value:        2.830e-08
==================================================================================================
                 coef    std err          z      P>|z|      [95.0% Conf. Int.]
```

```
##Nato_And_EU Not_In_Nato_Not_In_EU

recode4 = {'Nato_And_EU':'Nato_And_EU', 'Not_In_Nato_Not_In_EU': 'Not_In_Nato_Not_In_EU'}

datasub2['COMP1v4']= datasub2['NATO_EU_MEMBERSHIP'].map(recode4)

ct4=pandas.crosstab(datasub2['polityscore_cat_democracy'], datasub2['COMP1v4'])

print (ct4)

# column percentages

colsum=ct4.sum(axis=0)

colpct=ct4/colsum

print(colpct)

##

print ('chi-square value, p value, expected counts')

cs4= scipy.stats.chi2_contingency(ct4)

print (cs4)

print (cs4)

## (10.152916666666666, 0.0014407311825336937, 1L, array([[ 14.28571429,  10.71428571],

##       [ 5.71428571,   4.28571429]]))

##



##Not_In_Nato_In_EU Not_In_Nato_Not_In_EU

recode5 = {'Not_In_Nato_In_EU': 'Not_In_Nato_In_EU', 'Not_In_Nato_Not_In_EU': 'Not_In_Nato_Not_In_EU'}

datasub2['COMP2v3']= datasub2['NATO_EU_MEMBERSHIP'].map(recode5)

ct5=pandas.crosstab(datasub2['polityscore_cat_democracy'], datasub2['COMP2v3'])

print (ct5)

# column percentages

colsum=ct5.sum(axis=0)

colpct=ct5/colsum

print(colpct)

##

print ('chi-square value, p value, expected counts')
```

post hoc test to show the pairs of groups that the means that are significantly different.

Between the pairs we can find no evidence of statistically significant difference in means (from the summary output we fail to reject the null hypothesis that there is no difference in means).

```
import statsmodels.stats.multicomp as multi

mc1 = multi.MultiComparison(dataanovatestdf['incomeperperson'],dataanovatestdf['NATO_EU_MEMBERSHIP'])

res1 = mc1.tukeyhsd() ##tkeys honestly different test

print(res1.summary())

        Multiple Comparison of Means - Tukey HSD,FWER=0.05
=============================================================================
    group1          group2        meandiff    lower     upper    reject
-----------------------------------------------------------------------------
 Nato_And_EU       Nato_Not_In_EU   4149.6812 -27680.6426 35980.0049 False
 Nato_And_EU       Not_In_Nato_In_EU   7199.6882 -19850.8854 34250.2618 False
 Nato_And_EU       Not_In_Nato_Not_In_EU 3622.1337 -16227.5606 23471.828  False
 Nato_Not_In_EU    Not_In_Nato_In_EU    3050.007 -34462.3892 40562.4033 False
 Nato_Not_In_EU    Not_In_Nato_Not_In_EU -527.5475 -33230.0964 32175.0014 False
Not_In_Nato_In_EU  Not_In_Nato_Not_In_EU -3577.5545 -31649.2614 24494.1524 False
-----------------------------------------------------------------------------
```

Appendix 1: Code to run analysis all code available here

```
##week 1

##filter european countries

##filter out NA's

##select columns
```

polityscore_cntre
89.407 -0.322
147.951

femaleemployrate
35.390 1.926
138.126

armedforcesrate_
361.136 0.29
820.400

internetuserate_c
19.150 15.584
336.296

=============
=============
=============

The intercept is i
centred values, t
6884.5758, which
average of all the
income per perso
particular country

Other values wor
omnibus test, the
both skew and ku
null hypothesis th
normal. As we ob
0.005 we can rej
that the residuals
distributed. This i
Jarque Bera test,
Bera test uses sk
kurtosis (K), the r
the distribution is
0.000897, is sma
hypothesis that th
normal. We can a
by visualizing the
residuals.

Analysis of norr using qq plot

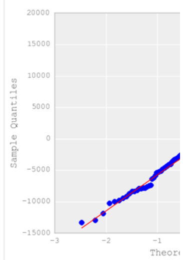


Figure qqplot of r

We can see from
residuals except
appear to lie alon
So here we can s
distributions do n
normally distribut

Standardized re

——————————————————————————

Intercept 0.4541 0.215 2.117 0.034 0.034 0.875

incomeperperson_centred 3.998e-06 4.59e-05 0.087 0.931 -8.59e-05 9.39e-05

urbanrate_centred 0.0016 0.013 0.125 0.901 -0.024 0.027

internetuserate_centred 0.0449 0.015 3.023 0.003 0.016 0.074

armedforcesrate_centred -0.6430 0.193 -3.338 0.001 -1.020 -0.265

femaleemployrate_centred -0.0139 0.015 -0.934 0.350 -0.043 0.015

==

Odds ratio Model 2

The odds ratio is shown below:

2.5% 97.5% OR

Intercept 1.034187 2.398046 1.574811

incomeperperson_centred 0.999914 1.000094 1.000004

urbanrate_centred 0.976481 1.027406 1.001620

internetuserate_centred 1.015899 1.076728 1.045871

armedforcesrate_centred 0.360439 0.766844 0.525738

femaleemployrate_centred 0.957784 1.015399 0.986171

Evidence for confounding

The original analysis looked at the relationship between the democracy response variable (1 is a democracy, 0 is not a democracy) and armed forces rate, the explanatory variable.

However when using multiple explanatory variables, analysis found that armed forces rate is still statistically significant This would be evidence for armed forces rate not being an example of a confounded with the other explanatory variables.

Conclusion – Whether the results gathered support the hypothesis for the association between your

cs5= scipy.stats.chi2_contingency(ct5)

print (cs5)

"A chi squared test was used to determine if NATO and EU membership (or non membership) had an effect on a country being a democracy (democracy) or a non-democracy (anocracy or autocracy). All code is available here: https://github.com/brennap3/Gapminder/blob/master/Gapminder_Analysis_2015.pyAfter calculating the different categories and sub-setting for European countries. A chi squared test is run. The values returned by the test are (appendix 1 shows the code used to run this test):· Chi-squared statistic13.636363636363637· P value 0.0034443294821406593·
Our alpha value is 0.05 and are p value is 0.0034443294821406593, this is below the critical value so we reject the null hypothesis (that the relative proportions of one variable are independent of the second variable) that there is no difference between groups. There is a statistically difference between groups (that the relative proportions of one variable are associated with the second variable).The difference between the groups is shown
below NATO_EU_MEMBERSHIP Nato_And_EU Nato_Not_In_EU Not_In_Nato_In_EU \polityscore_cat_democracy Democracy 19 3 5
Not Democracy 1 1 1
NATO_EU_MEMBERSHIP Not_In_Nato_Not_In_EU polityscore_cat_democracy Democracy 6 Not Democracy
9 Next we run a series of post hoc tests (these are shown in appendix 2), the corrected p value which we reject the null hypothesis is equal to 0.05/3 or 0.016666 (as we are conducting 3 tests). The 3 different groups we are comparing are:· "Nato_And_EU", "Nato_Not_In_EU" (p value: 0.94643404330274994)·
"Nato_And_EU"
"Not_In_Nato_Not_In_EU" (p value: 0.0014407311825336937)·
"Not_In_Nato_In_EU"
"Not_In_Nato_Not_In_EU"(p value: 0.18931317392613722)So the only statistically significant difference between groups identified by the post hoc tests is "Nato_And_EU"
"Not_In_Nato_Not_In_EU" (p value: 0.0014407311825336937). We can say that the Chi-Square Test of

##run ANOVA

##

##checking coloumns

##data.columns.values

##checking values

##data['European']

dataanovatestdf=data[['country','incomeperperson','NATO_EU_MEMBERSHIP']][data.European=='Europe']

##dataanovatestdf

model1 = smf.ols(formula='incomeperperson ~ C(NATO_EU_MEMBERSHIP)', data=data)

results1 = model1.fit()

print(results1.summary())

##F-statistic: 8.026

##Prob (F-statistic): 4.66e-05

p value less than 0.05 a difference in variance in different groups

print ('means for incomme per person for different groups')

meansincomeperpersn= dataanovatestdf.groupby('NATO_EU_MEMBERSHIP').mean()

print (meansincomeperpersn)

##lets display it using boxplots

print ('variances for incomme per person for different groups')

varianceincomeperpersn= dataanovatestdf.groupby('NATO_EU_MEMBERSHIP').var()

print (varianceincomeperpersn)

##

print ('standard deviations for incomme per person for different groups')

standarddeviationincomeperpersn= dataanovatestdf.groupby('NATO_EU_MEMBERSHIP').std()

print (standarddeviationincomeperpersn)

##boxplot

ggplot(dataanovatestdf, aes(x='incomeperperson', y='NATO_EU_MEMBERSHIP')) + geom_boxplot() +\

The standardized
below. As there a
points lying great
deviations indicat
in the data. The s
plot shows that m
observations lie c
that the model giv
fit of the data.

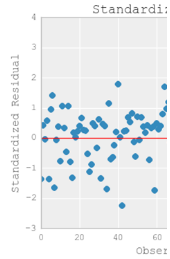


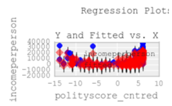
Figure standardiz
evidence that the
and is a poor fit fo

Partial regressio

Partial regressio score

The residuals ver
in the upper right
funnel shape div
polity score. This
the model is poor
per person for co
polity scores.

The partial regres
attempts to show
(explanatory vari
on the response
person), controllin
explanatory varia
the relationship b
variable and the e
after controlling fo
explanatory varia
the plot to see if i
non-linear pattern
pattern, it meets t
assumption of ou
residuals are ran
the partial regress
from the line indic
of prediction erro



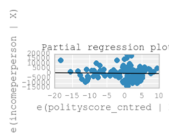


Figure Partial reg

primary explanatory variable (Democracy) and the primary response variable (armed forces rate).

In conclusion based on the results from the multiple models, particularly the second model, we can see from the extracted Odds ratios that armed forces rate was significantly associated with democracy such that countries with higher armed forces rate were significantly less likely to be a democracy (OR= 0.525738, 95% CI=0.360439-0.766844 , p=0.001). While internet use rate is also significantly associated with democracy such that countries with higher internet usage rates are more likely to be democracies (OR= 1.045871, 95% CI=1.015899 -1.076728 , p=0.003).

Appendix 1:

def polityscore_cat (row):

if (row['polityscore'] >=6 and row['polityscore'] <= 10) :

return 'Democracy'

elif (row['polityscore'] >=-5 and row['polityscore'] <= 5) :

return 'Anocracy'

elif (row['polityscore'] >=-10 and row['polityscore'] <= -6) :

return 'Autocracy'

else :

return 'NA'

def polityscore_cat_int (row):

if (row['polityscore_cat']=='Democracy') :

return 1

else :

return 0

Appendix 2 preprocessing and modelling code

####week 4 logistic rergression

##select the values of interest

##'polityscore_cat'

##'incomeperperson'

##'urbanrate'

##'internetuserate'

##'armedforcesrate'

datalogmodeltdf=data[['polityscore_cat', 'incomeperperson'

,'urbanrate'

Independence comparing frequencies of one categorical variable (Democracy and Non democracy) for different values of a second categorical variable (NATO EU membership, “Nato_And_EU” “Not_In_Nato_Not_In_EU”), we can say that alternate hypothesis holds true and that the relative proportions of the democracy variable is associated with the NATO EU membership, (“Nato_And_EU” “Not_In_Nato_Not_In_EU”) variable. Appendix 1cs1= scipy.stats.chi2_contingency(ct1)>>> print(cs1)(13.636363636363637, 0.0034443294821406593, 3L, array([[14.66666667, 2.93333333, 4.4 , 11.], [5.33333333, 1.06666667, 1.6 , 4.]])) Appendix2 ##Nato_And_EU Not_In_Nato_In_EU recode3 = {'Nato_And_EU':'Nato_And_EU', 'Not_In_Nato_In_EU': 'Not_In_Nato_In_EU'}datasub2['COMP1v3']= datasub2['NATO_EU_MEMBERSHIP'].map(recode3) ct3=pandas.crosstab(datasub2['polityscore_cat_democracy'], datasub2['COMP1v3'])print (ct3) # column percentagescolsum=ct3.sum(axis=0)colpct=ct3/colsumprint(colpct)##print ('chi-square value, p value, expected counts')cs3= scipy.stats.chi2_contingency(ct3)print (cs3) ##0.10909090909090913 0.74118150587360399 ##Nato_And_EU Not_In_Nato_Not_In_EU recode4 = {'Nato_And_EU':'Nato_And_EU', 'Not_In_Nato_Not_In_EU': 'Not_In_Nato_Not_In_EU'}datasub2['COMP1v4']= datasub2['NATO_EU_MEMBERSHIP'].map(recode4) ct4=pandas.crosstab(datasub2['polityscore_cat_democracy'], datasub2['COMP1v4'])print (ct4) # column percentagescolsum=ct4.sum(axis=0)colpct=ct4/colsumprint(colpct)##print ('chi-square value, p value, expected counts')cs4= scipy.stats.chi2_contingency(ct4)print (cs4)print (cs4)##(10.152916666666666, 0.0014407311825336937, 1L, array([[14.28571429, 10.71428571],## [5.71428571, 4.28571429]]))## ##Not_In_Nato_In_EU Not_In_Nato_Not_In_EU recode5 = {'Not_In_Nato_In_EU': 'Not_In_Nato_In_EU', 'Not_In_Nato_Not_In_EU': 'Not_In_Nato_Not_In_EU'}datasub2['COMP2v3']= datasub2['NATO_EU_MEMBERSHIP'].map(recode5) ct5=pandas.crosstab(dat

xlab(“incomeperperson”) + ylab(“ Nato EU membership status”) + ggtitle(“Boxplot for income per person for gapminder data for European countries using categorical variable based on Nato Eu membership”)

##

##run your post hoc tests

import statsmodels.stats.multicomp as multi

mc1 = multi.MultiComparison(dataanovatestdf['incomeperperson'], dataanovatestdf['NATO_EU_MEMBERSHIP'])

res1 = mc1.tukeyhsd() ##tkeys honestly different test

print(res1.summary())

##must be performed after

##cant run pairwise

Regression Modeling in Practice Week 1 Assignment

Sample

I have decided to use the Gapminder dataset, which uses a number of observational data collected from a number of dependable sources. Gapminder provides a number of different variables which describes 213 regions across the world (192 UN countries, plus additional 21 regions). Each indicator was collected by different source authority and the data is then collated by Gapminder. Each variable is collected by Region.The study will be carried out by individual country and region.These include Polity, World Bank, World Economic Forum, transparency and the United Nations. There is no specific methodology of collection as each data source has its own methodology and collection method. Gapminder serves as a collator of this data by different sources. The potential variables to be used are listed below. The dataset used is Gapminder 2015 (Date: 08/11/2015).

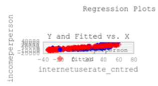
Summary statistics are listed below showing the range of the data collected:

score centred va

Partial regressic usage

The residuals ver plot in the upper shows no obviou that over the rang there is no obviou predictive power internet usage.

The partial regres attempts to show (explanatory vari score has on the (income per pers other explanatory shows the relatio response variable variable after con explanatory varia the plot to see if i non-linear pattern pattern, it meets t assumption of ou indeed show a lin residuals random partial regression observations lie f indicating a large error for internet u



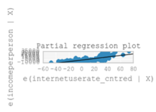


Figure Partial reg internet usage ce

Leverage plot

Next we look at le is a means of qua and independent its average. Point can have a profo estimates of regr Points with stand greater than 2 or considered outlie that 112 and 118 Leverage. observ leverage but does

,'internetuserate','femaleemployrate'

,'armedforcesrate']].dropna()

datalogmodeltdfnona=datalogmodeltdf[(data.polityscore_cat!='NA')]

datalogmodeltdf.dtypes

##build logistic model

datalogmodeltdfnona['polityscore_cat']=datalogmodeltdfnona['polityscore_cat'].astype(str)

datalogmodeltdfnona.dtypes

datalogmodeltdfnona=datalogmodeltdfnona.reset_index()

del datalogmodeltdfnona['index']

datalogmodeltdfnona

datalogmodeltdfnonav1=datalogmodeltdfnona[['polityscore_cat','incomeperperson','urbanrate'

,'internetuserate','femaleemployrate'

,'armedforcesrate'

]].dropna()

##recode variables

def polityscore_cat_int (row):

if (row['polityscore_cat']=='Democracy') :

return 1

else :

return 0

##recode if democracy 1 else (it its anocracy or autocracy)

##calculate the age of NATO countries

##data['Years_In_Nato'] = data.apply (lambda row: AGE_YEARS (row),axis=1)

datalogmodeltdfnonav1['polityscore_cat_int'] = datalogmodeltdfnonav1.apply (lambda row: polityscore_cat_int (row),axis=1)

#####Pre-processing data

datalogmodeltdfnonav1_centered = preprocessing.scale(datalogmodeltdfnonav1[['incomeperperson','urbanrate','internetuserate','femaleemployrate','armedforcesrate']], with_mean=True, with_std=False) ##corrected this had

asub2['polityscore_cat_democracy'], datasub2['COMP2v3'])print (ct5) # column percentagescolsum=ct5.sum(axis=0)colpct=ct5/colsumprint(colpct)##print ('chi-square value, p value, expected counts')cs5= scipy.stats.chi2_contingency(ct5)print (cs5) "

Week 3 Making Data Management Decisions

Overview:

I decided to create some new categorical variables based upon region (European) and membership of institutions such as the EU and Nato. From these I created some more categorical variables based on the ones I added these can be referred to as secondary categorical variables. One of the secondary variables created was if a country is a member of Nato and a member of EU.

Frequency tables and Contingency tables

Nato and EU membership contingency tables

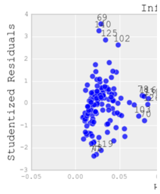
Not_In_Nato_And_In_EU_Member	6
Nato_And_EU_Member	20
Not_In_Nato_Not_In_EU	181
Nato_And__Not_In_EU	6

This frequency table was created by adding a variable called 'Eu_Member' (through a lambda function), 'Is_Nato_Country' (this was created by merging two dataframe together one with gapminder data and the other with Nato data , which included country and join data) an from these creating a secondary variable called NATO_EU_MEMBERSHIP (through a lambda function). The logic for this was in pseudo code (the actual function is EU_NATO in script).

```
if Nato_Membership ==
'Nato_Member' and EU_Membership==
'EU' :
    return 'Nato_And_EU_Member'
elif Nato_Membership ==
'Nato_Member' and EU_Membership
== 'Not-In-EU':
    return 'Nato_And__Not_In_EU'
elif Nato_Membership !=
'Nato_Member' and EU_Membership==
'EU' :
```

```
      incomeperperson  alcconsumption
 armedforcesrate breastcancerper100th

count     190.000000      187.000000
 164.000000         173.000000
mean     8740.966076        6.689412
    1.444016         37.402890
std     14262.809083        4.899617
 1.709008          22.697901
min       103.775857        0.030000
 0.000000          3.900000
25%       748.245151        2.625000
 0.480907          20.600000
50%      2553.496056        5.920000
    0.930638         30.000000
75%      9379.891165        9.925000
    1.611383         50.300000
max    105147.437697       23.010000
     10.638521        101.100000

      co2emissions  femaleemployrate
hivrate  internetuserate

count  2.000000e+02       178.000000
 147.000000       192.000000
mean   5.033262e+09        47.549438
 1.935442        35.632716
std    2.573812e+10        14.625743
 4.376727        27.780285
min    1.320000e+05        11.300000
 0.060000         0.210066
25%    3.484617e+07        38.725000
 0.100000         9.999604
50%    1.859018e+08        47.549999
 0.400000        31.810121
75%    1.846084e+09        55.875000
 1.300000        56.416046
max    3.342209e+11        83.300003
25.900000        95.638113

      lifeexpectancy  oilperperson
 polityscore  relectricperperson

count     191.000000     63.000000
161.000000        136.000000
mean       69.753524      1.484085
3.689441        1173.178995
std         9.708621      1.825090
6.314899        1681.440173
min        47.794000      0.032281
-10.000000          0.000000
25%        64.447000      0.532541
 -2.000000        203.652109
50%        73.131000      1.032470
6.000000         597.136436
75%        76.593000      1.622737
9.000000        1491.145249
max        83.394000     12.228645
 10.000000      11154.755033

      suicideper100th  employrate
urbanrate  Year_Joined_Nato

count      191.000000  178.000000
 203.000000       27.000000

mean        9.640839   58.635955
56.769360       1973.074074
std         6.300178   10.519454
```

outlier (its standa
is less than 2 anc
value is close to (



Summary Statis

>>> print (mreg1l

C

Results

============
============
============

Dep. Variable:
R-squared:

Model:
squared:

Method:
statistic:

Date: Su
(F-statistic):

Time:
Likelihood:

No. Observations

Df Residuals:
29

Df Model:

Covariance Type

============
============
============
=======

t P>|t|

____
____

Intercept
14.351 0.00(
7832.993

polityscore_cntre
89.407 -0.322
147.951

femaleemployrate
35.390 1.926
138.126

wrong version of code had True and False in qoutes now its in correct format

##cast it as a dataframe

datalogmodeltdfnonav1_centered_df = pd.DataFrame(datalogmodeltdfnonav1_centered)

set the columns

datalogmodeltdfnonav1_centered_df.columns= ['incomeperperson_centred','urbanrate_centred','internetuserate_centred','femaleemployrate_centred','armedforcesrate_centred']

check the count

datalogmodeltdfnonav1_centered_df.count()

##all look sfine

##data

##data 3 is our second subset we will use to do some analysis

##concatanate once the indexs are reset

datalogmodeltdfnonav1_centered_df_cntred = pd.concat([datalogmodeltdfnonav1['polityscore_cat_int'], datalogmodeltdfnonav1_centered_df], axis=1)

datalogmodeltdfnonav1_centered_df_cntred.columns.values

##

preprocessing ended

lreg1 = smf.logit(formula='polityscore_cat_int ~ armedforcesrate_centred',data = datalogmodeltdfnonav1_centered_df_cntred).fit()

print (lreg1.summary())

##odds ratio

print np.exp(lreg1.params)

##little or no effect

params = lreg1.params

conf = lreg1.conf_int()

conf['OR'] = params

conf.columns = ['2.5%', '97.5%', 'OR']

print np.exp(conf)

return 'Not_In_Nato_And_In_EU_Member'
else :
return 'Not_In_Nato_Not_In_EU'

distribution of years in NATO for Nato Countries

```
-1     187
6       2
11      6
16      2
33      1
60      1
63      2
66     12
```

** -1 indicates country is not a member

The ages were calculated by subtracting the year a country joined NATO from the current year. The function used to do this is the AGE_YEARS function. it should be noted if a country has not joined NATO a value of -1 is returned. -1 Was chosen as it is a negative number and cannot relate to age.

distribution of politly score for EU countries

```
10    17
9      5
8      2
NaN    2
```

Note 2 are blank when compared to the whole world, its quite different note 52 countries worldwide are blank.

frequency table of politly scores worldwide

frequency table of politly scores

```
NaN    52
10    33
8     19
9     15
7     13
-7    12
6     10
5      7
-3     6
0      6
-4     6
-2     5
-1     4
-9     4
4      4
2      3
1      3
-6     3
-5     2
-10    2
3      2
-8     2
```

Plots -Using our new variables

```
23.844933       26.939999
min        0.201449   32.000000
10.400000       1949.000000
25%          4.988449   51.225000
36.830000       1949.000000
50%          8.262893   58.699999
57.940000       1952.000000
75%         12.328551   64.975000
74.210000       2004.000000
max         35.752872   83.199997
 100.000000       2009.000000
```

```
      Years_In_Nato
count     213.000000
mean        4.568075
std        17.423994
min        -1.000000
25%        -1.000000
50%        -1.000000
75%        -1.000000
max        67.000000
```

Note Years in Nato is calculated based on Nato's membership data.

Transparency data will have to be married to the Gapminder download. Other data not listed if needed will have to be scraped from the relevant website and added.

Procedure

The data contains a number indicators, all collected with different methodology and different sources (as I mention before). Currently I don't know which one will be incorporated in my work, I will only list them below and their sources. The sources and the variables are all listed in accompanying codebook for the dataset. The dataset was collected by Gapminder in 2015. As Gapminder is a collator of data, they have collated this data from numerous sources using different methodologies. The dataset was originally created by Gapminder to serve as an educational resource to allow a fact based understanding of the world.

Measures

Variables considered for this study are:

Variables:

The variables (and the sources) I plan to use are:

country
incomeperperson
alcconsumption
armedforcesrate
breastcancerper100th
co2emissions
femaleemployrate
hivrate
internetuserate
lifeexpectancy
oilperperson

```
armedforcesrate_
 361.136     0.29
  820.400

internetuserate_c
19.150     15.584
336.296

=============
=============
=============

Omnibus:
Durbin-Watson:

Prob(Omnibus):
Jarque-Bera (JB)

Skew:
                0.00

Kurtosis:
No.

=============
=============
=============
```

Warnings:

[1] Standard Erro
covariance matrix
correctly specifie

Conclusion

There still appear
our model and se
of the assumption
particularly norma
of our residuals a
the explanatory v
break the assump
was revealed thr
regression plots.

Besides the probl
only internet usag
employee rate (th
at the 0.05 level)
statistically signifi
income per perso
appears to have
significant influen
does polity score

Confounding va

The original analy
relationship betw
(polity score) and
explanatory varia
multivariate analy
longer a statistica
only internet use
significant. This w
a confounding va

In relation to stati
variable is an add
statistical model t

```
lreg3 =
smf.logit(formula='polityscore_cat_int ~
incomeperperson_centred+urbanrate_c
entred+internetuserate_centred+armedf
orcesrate_centred+femaleemployrate_
centred',data =
datalogmodeltdfnonav1_centered_df_c
ntred).fit()

print (lreg3.summary())

##odds ratio

params = lreg3.params

conf = lreg3.conf_int()

conf['OR'] = params

conf.columns = ['2.5%', '97.5%', 'OR']

print np.exp(conf)
```

Week 2 Python Program

1 note

Dataset: Gapminder

Program Code:

```
# -*- coding: utf-8 -*-
"""
This is my Gapminder script file.
it reads in data
changes the type to numerics
and does some univariate analaysis
using
analytic visuals and frequency tables
"""
import os
import pandas
import numpy
import sklearn
import matplotlib
import matplotlib.pyplot as plt
##give the path of our folder
##set the path to wherever you
downloaded the dataset
apath='C:\Users\Peter\Desktop\Gapmi
nder'
print(apath)
os.chdir('C:\Users\Peter\Desktop\Gap
minder')
##check the directory has changed
os.getcwd()
##read in the file
data =
pandas.read_csv('gapminder.csv',
low_memory=False)
##check the first 10 columns
##equivalent of r head
data.head(10)
##
columnnames_list_=list(data.columns.v
alues)
## print the list
print(columnnames_list_)

data.dtypes
"""
country                object
incomeperperson        object
```

This can be used to create a faceted grid scatter plot of all the different categories of NATO and EU membership

NATO and EU m

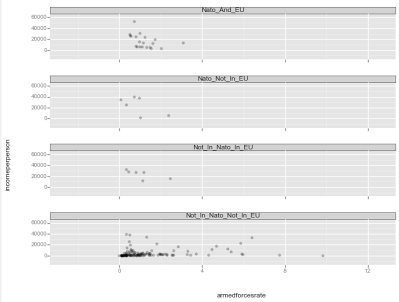
NATO and Non EU

Non NATO and EU

Non NATO and NON EU

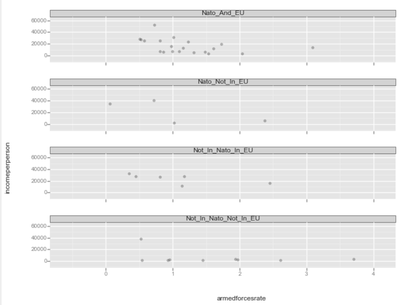
Whole world

You can see from the facted plot below that for all the different categories their doe not appear to be a relationship between the variables, in the first three plots (EU and NATO membership) countries with larger incomes per person have lower armed forces rates. For countries not in NATO or the EU the trend seems to be reversed countries with low incomes per person have low armed forces rates.



Europe

If we repeat just for Europe (where the countries are European), we see something else, for NATO or EU countries countries with higher Incomes have lower armed forces rates. While for non NATO and NON EU countries there does not appear to be any relationship between income per person and armed forces rate



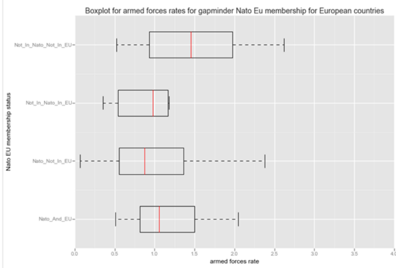
Script

```
# -*- coding: utf-8 -*-
"""
Spyder Editor

This is my Gapminder script file.
it reads in data
changes the type to numerics
and does some univariate analaysis
using
```

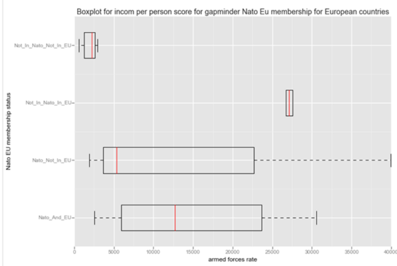
polityscore
relectricperperson
suicideper100th
employrate
urbanrate

The aim of the study is to examine the relationship between how the transparency, Polity score, member of economic or military communities/groups as explanatory variables, armed forces rate for response variables income per person and poverty %.

Below we see a boxplot for armed forces for European countries and how Nato And EU membership affects the armed forces rate. You can see that countries not in Nato and not in the EU have higher armed forces rates than those that are or are in both.



We can also see that income per person is much higher for EU and Nato members. EU members and Nato members have higher incomes person. However countries that are in Nato and not in EU have incomes that are are lower than those that are in the EU.



(positive or negat
both the explanat
variable.

A spurious relatio
association betw
variable and a res
has been assess
because the estir
the confounding v
assessment is du
bias. This is what

When we measu
between :

1. Incomer pe
score (correlation
0.291390226361

2. Income per
use rate (correlat
0.812957773808

3. Polity score
rate. (correlation
0.371956208119

We see all combi
correlation this w
further evidence
variable as intern
originally omiited
simple linear regr
weeks analysis),
with income per p
variable) and poli
explanatory varia

Code

datamv4=datamv
mv4['incomeperp

pearsonr(datamv
atamv4['polityscc

##first is correlati
values

##(0.2913902263
0.000359202528

##

pearsonr(datamv
atamv4['internetu

##first is correlati
values

##(0.8129577738
1.241632044123

pearsonr(datamv
datamv4['internet

##first is correlati
values

##(0.3719562081
3.787845602079

KOUdy��9e�

```
alcconsumption          object
armedforcesrate         object
breastcancerper100th    object
co2emissions            object
femaleemployrate        object
hivrate                 object
internetuserate         object
lifeexpectancy          object
oilperperson            object
polityscore             object
relectricperperson      object
suicideper100th         object
employrate              object
urbanrate               object
dtype: object
"""
##lets convert the data to  numeric
data['incomeperperson'] =
data['incomeperperson'].convert_object
s(convert_numeric=True)
data['alcconsumption'] =
data['alcconsumption'].convert_objects(
convert_numeric=True)
data['armedforcesrate'] =
data['armedforcesrate'].convert_objects
(convert_numeric=True)
data['breastcancerper100th'] =
data['breastcancerper100th'].convert_o
bjects(convert_numeric=True)
data['co2emissions'] =
data['co2emissions'].convert_objects(c
onvert_numeric=True)

data['femaleemployrate'] =
data['femaleemployrate'].convert_object
s(convert_numeric=True)
data['hivrate'] =
data['hivrate'].convert_objects(convert_
numeric=True)
data['internetuserate'] =
data['internetuserate'].convert_objects(
convert_numeric=True)
data['lifeexpectancy'] =
data['lifeexpectancy'].convert_objects(c
onvert_numeric=True)
data['oilperperson'] =
data['oilperperson'].convert_objects(co
nvert_numeric=True)

data['polityscore'] =
data['polityscore'].convert_objects(conv
ert_numeric=True)
data['relectricperperson'] =
data['relectricperperson'].convert_objec
ts(convert_numeric=True)
data['suicideper100th'] =
data['suicideper100th'].convert_objects(
convert_numeric=True)
data['employrate'] =
data['employrate'].convert_objects(con
vert_numeric=True)
data['urbanrate'] =
data['urbanrate'].convert_objects(conve
rt_numeric=True)
## or use the describe function
##this gives us some summary
inforation about our data
##this will give summary info for each
analytic visuals and frequency tables
"""
import os
import pandas
import numpy
import sklearn
import matplotlib
import matplotlib.pyplot as plt
import sys; print(sys.path)
from seaborn import *
from ggplot import *

##give the path of our folder
##set the path to wherever you
downloaded the dataset
apath='C:\Users\Peter\Desktop\Gapmi
nder'
print(apath)
os.chdir('C:\Users\Peter\Desktop\Gap
minder')
##check the directory has changed
os.getcwd()
##read in the file
data =
pandas.read_csv('gapminder.csv',
low_memory=False)
##check the first 10 columns
##equivalent of r head
data.head(10)
##
columnnames_list_=list(data.columns.v
alues)
## print the list
print(columnnames_list_)

data.dtypes
"""
country                 object
incomeperperson         object
alcconsumption          object
armedforcesrate         object
breastcancerper100th    object
co2emissions            object
femaleemployrate        object
hivrate                 object
internetuserate         object
lifeexpectancy          object
oilperperson            object
polityscore             object
relectricperperson      object
suicideper100th         object
employrate              object
urbanrate               object
dtype: object
"""
##lets convert the data to  numeric
data['incomeperperson'] =
data['incomeperperson'].convert_object
s(convert_numeric=True)
data['alcconsumption'] =
data['alcconsumption'].convert_objects(
convert_numeric=True)
data['armedforcesrate'] =
data['armedforcesrate'].convert_objects
(convert_numeric=True)
data['breastcancerper100th'] =
data['breastcancerper100th'].convert_o
bjects(convert_numeric=True)
```

Appendix 1 Cod

```
###############
###Week 3
###########

data.columns.val
##only select not
 score
datamv1=data[pa
yscore'])&pandas
mployrate'])&pan
dforcesrate'])&pa
netuserate'])].cop
#carry out a chec
Kingdom
datamv1['politysc
== 'United Kingdo
##all looks ok
##now subset the
polityscore and c
datam2=datamv1
mployrate','armed
erate']]
##
## centre the poli
mx_centered =
preprocessing.sc
re','femaleemploy
,'internetuserate']
with_std=False) #
wrong version of
False in qoutes n
format
##cast it as a dat
mx_centered_df =
pd.DataFrame(m
columns=datam2
## check the cou
mx_centered_df.
##all look sfine
##data
##data 3 is our se
use to do some a
## it consists of c
person and polity
datam3=datamv1
person','polityscc
'armedforcesrate
##reset the index
datam3=datam3.r
del datam3['index
##
#data3['polityscor
##concatanate or
reset
datamv4 = pd.co
mx_centered_df],
datamv4.columns
'country', 'income
'polityscore', 'fem
      'armedforces
'polityscore',
      'femaleemplo
'armedforcesrate
```

row
##this is handy for looking at descriptive univariate analysis

pandas.DataFrame.describe(data)
##subsetting the data
##lets take a subset of data for northern european countries
euro_list=('Belgium','France','Netherlands','Ireland','United Kingdom','Germany','Denmark','Sweden','Norway','Finland')
##subset by getting the index of the countries in above list
##and then getting the data at these indexes
subset_northeastern_europe=data[data['country'].isin(euro_list)]
##lets check the subset operation
subset_northeastern_europe.head(10)
pandas.DataFrame.describe(subset_northeastern_europe)
##as you can see the dataa is very different for North Western Europe

income per person
print(“frequency table of incomeperperson”)
p1=data['incomeperperson'].value_counts(sort=False, normalize=True,dropna=False)
print(p1)
##this does not tell me much lets plot the hsitogram
##lets create some categorical varaibles
data['incomeperperson']

bins = [0, 1000, 5000, 10000, 20000,50000,200000]
group_names = ['Very Low Income,0-1000', 'Low Income,1000-5000', 'Okay Income,5000-10000', 'Good Income,10000-20000','Great Income,20000-50000','50,000-200,000']

categories = pandas.cut(data['incomeperperson'], bins, labels=group_names)
data['categories'] = pandas.cut(data['incomeperperson'], bins, labels=group_names)
#print('new categorical variables based on the income per person')
pandas.value_counts(data['categories'])
##from our ouptu we can see that the vast najority of countries
##surveyed are in the low income or very low income category
“'
Low Income,1000-5000 61
Very Low Income,0-1000 54
Okay Income,5000-10000 28
Great Income,20000-50000 26
Good Income,10000-20000 17

data['co2emissions'] = data['co2emissions'].convert_objects(convert_numeric=True)

data['femaleemployrate'] = data['femaleemployrate'].convert_objects(convert_numeric=True)
data['hivrate'] = data['hivrate'].convert_objects(convert_numeric=True)
data['internetuserate'] = data['internetuserate'].convert_objects(convert_numeric=True)
data['lifeexpectancy'] = data['lifeexpectancy'].convert_objects(convert_numeric=True)
data['oilperperson'] = data['oilperperson'].convert_objects(convert_numeric=True)

data['polityscore'] = data['polityscore'].convert_objects(convert_numeric=True)
data['relectricperperson'] = data['relectricperperson'].convert_objects(convert_numeric=True)
data['suicideper100th'] = data['suicideper100th'].convert_objects(convert_numeric=True)
data['employrate'] = data['employrate'].convert_objects(convert_numeric=True)
data['urbanrate'] = data['urbanrate'].convert_objects(convert_numeric=True)
or use the describe function
##this gives us some summary inforation about our data
##this will give summary info for each row
##this is handy for looking at descriptive univariate analysis

pandas.DataFrame.describe(data)
##subsetting the data
##lets take a subset of data for northern european countries
euro_list=('Belgium','France','Netherlands','Ireland','United Kingdom','Germany','Denmark','Sweden','Norway','Finland')
##subset by getting the index of the countries in above list
##and then getting the data at these indexes
subset_northeastern_europe=data[data['country'].isin(euro_list)]
##lets check the subset operation
subset_northeastern_europe.head(10)
pandas.DataFrame.describe(subset_northeastern_europe)
##as you can see the dataa is very different for North Western Europe

income per person
print(“frequency table of incomeperperson”)
p1=data['incomeperperson'].value_cou

##reset the colun
datamv4.columns
['country','income
e','femaleemployr
'internetuserate','
maleemployrate_
te_cntred','interne
print(datamv4)
##
##now reset the c
index
datamv4reidx=da
ntry')
datamv4reidx.col
##build rergessio
centred
build the multi
mreg1b = smf.ols
polityscore_cntre
cntred+armedforc
etuserate_cntred
print (mreg1b.sur

#Q-Q plot for nor

fig4=sm.qqplot(m

# simple plot of re
stdres=pandas.D
d_pearson)
plt.plot(stdres, 'o'
l = plt.axhline(y=0
plt.ylabel('Standa
plt.xlabel('Observ
plt.title('Standardi

# additional regre
import matplotlib.
pd.set_option('dis
'default')
fig2 = pltt.figure()
fig2 =
sm.graphics.plot_
b, "polityscore_c
print(fig2)

##
pd.set_option('dis
'default')
fig = pltt.figure()
fig =
sm.graphics.plot_
b, "internetusera
print(fig)

internetuserate_c

# leverage plot
fig3=sm.graphics
b, size=8)
print(fig3)

##change a colun
##you can see th
rpoblems

����T��

```
50,000-200,000        4
dtype: int64
"'

##boxplots for northern europe

incomeratedisteurope=subset_northeastern_europe['incomeperperson']
[(subset_northeastern_europe['incomeperperson'] >= 0)].values
plt.boxplot(incomeratedisteurope)

##boxplots for northern europe

incomeratedistworld=data['incomeperperson'][(data['incomeperperson'] >=
0)].values
plt.boxplot(incomeratedistworld)

##lets rerun the analysis for the northeren europe subset
##subset the data
categories_ne =
pandas.cut(subset_northeastern_europe['incomeperperson'], bins,
labels=group_names)
subset_northeastern_europe['categories'] =
pandas.cut(subset_northeastern_europe['incomeperperson'], bins,
labels=group_names)

pandas.value_counts(subset_northeastern_europe['categories'])

##all the north eastern european countries fall into the bracket
##Great Income,20000-50000    10
##its a very homogonous group
"'
Out[63]:
Great Income,20000-50000    10
50,000-200,000        0
Good Income,10000-20000     0
Okay Income,5000-10000      0
Low Income,1000-5000        0
Very Low Income,0-1000      0
dtype: int64
"'
##now lets do the same for:
#armedforcesrate         object
#polityscore             object

print("frequency table of armedforcesrate")
p2=data['armedforcesrate'].value_counts(sort=False,
normalize=True,dropna=False)
print(p2)
##23% of data is NaN
##only really usefull statistic
armedforcesrate=data['armedforcesrate'][(data['armedforcesrate'] >= 0)].values
bins=100
plt.hist(armedforcesrate, bins,
normed=True, color="#F08080",
alpha=.5);
##the armed forces rates density
nts(sort=False,
normalize=True,dropna=False)
print(p1)
##this does not tell me much lets plot the hsitogram
##lets create some categorical varaibles
data['incomeperperson']

bins = [0, 1000, 5000, 10000,
20000,50000,200000]
group_names = ['Very Low Income,0-1000', 'Low Income,1000-5000', 'Okay
Income,5000-10000', 'Good
Income,10000-20000','Great
Income,20000-50000','50,000-200,000']

categories =
pandas.cut(data['incomeperperson'],
bins, labels=group_names)
data['categories'] =
pandas.cut(data['incomeperperson'],
bins, labels=group_names)
#print('new categorical variables based on the income per person')
pandas.value_counts(data['categories']
)
##from our ouptu we can see that the vast najority of countries
##surveyed are in the low  income or very low income category
"'
Low Income,1000-5000        61
Very Low Income,0-1000      54
Okay Income,5000-10000      28
Great Income,20000-50000    26
Good Income,10000-20000     17
50,000-200,000        4
dtype: int64
"'

##lets rerun the analysis for the northeren europe subset
##subset the data
categories_ne =
pandas.cut(subset_northeastern_europe['incomeperperson'], bins,
labels=group_names)
subset_northeastern_europe['categories'] =
pandas.cut(subset_northeastern_europe['incomeperperson'], bins,
labels=group_names)

pandas.value_counts(subset_northeastern_europe['categories'])

##all the north eastern european countries fall into the bracket
##Great Income,20000-50000    10
##its a very homogonous group
"'
Out[63]:
Great Income,20000-50000    10
50,000-200,000        0
Good Income,10000-20000     0
Okay Income,5000-10000      0
Low Income,1000-5000        0
Very Low Income,0-1000      0
```

```
distribution is shown
##heavy tailed distribution
##alternatively as a boxplot

plt.boxplot(armedforcesrate)
#alternatively as a violin plot
plt.violinplot(armedforcesrate)

##look at the politly scores
print(“frequency table of politly scores”)
p3=data['polityscore’].value_counts(sort=False, normalize=True,dropna=False)
print(p3)
##
“’
frequency table of politly scores
NaN    0.244131
0     0.028169
1     0.014085
2     0.014085
3     0.009390
4     0.018779
5     0.032864
6     0.046948
7     0.061033
8     0.089202
9     0.070423
10    0.154930
-1     0.018779
-10    0.009390
-9     0.018779
-8     0.009390
-7     0.056338
-6     0.014085
-5     0.009390
-4     0.028169
-3     0.028169
-2     0.023474
”’

##again this does not tell us much
##lets create the

politlyscores=data['polityscore’]
[(data['polityscore’] >= -11)].values
#violin plot
plt.violinplot(politlyscores)
##most dense distribution in range 5-10
```

Findings:

The 3 variables chosen from gapminder are:

- Income rate
- Armed forces rate
- Politly score

Their distributions where examined by frequency tables and analytical visualizations to show the distributions. The type of visualizations used where:

- Violin Plots
- Boxplots
- Histograms

It does not make much sense to show the frequency table (the code is included to create these), so instead

```
dtype: int64
”’
##now lets do the same for:
#armedforcesrate         object
#polityscore             object

print(“frequency table of armedforcesrate”)
p2=data['armedforcesrate’].value_counts(sort=False, normalize=True,dropna=False)
print(p2)
##23% of data is NaN
##only really usefull statistic
armedforcesrate=data['armedforcesrate’][(data['armedforcesrate’] >= 0)].values
bins=100
plt.hist(armedforcesrate, bins, normed=True, color=“#F08080”, alpha=.5);
##the armed forces rates density distribution is shown
##heavy tailed distribution
##alternatively as a boxplot

plt.boxplot(armedforcesrate)
#alternatively as a violin plot
plt.violinplot(armedforcesrate)

##look at the politly scores
print(“frequency table of politly scores”)
p3=data['polityscore’].value_counts(sort=True, dropna=False)
print(p3)
##
“’
frequency table of politly scores
NaN    0.244131
0     0.028169
1     0.014085
2     0.014085
3     0.009390
4     0.018779
5     0.032864
6     0.046948
7     0.061033
8     0.089202
9     0.070423
10    0.154930
-1     0.018779
-10    0.009390
-9     0.018779
-8     0.009390
-7     0.056338
-6     0.014085
-5     0.009390
-4     0.028169
-3     0.028169
-2     0.023474
”’

##again this does not tell us much
##lets create the

armedforcesrate=data['armedforcesrate’][(data['armedforcesrate’] >= 0)].values
bins=100
```

the distributions are shown or else categories are created by splitting the continuous variables.

Income per person

The frequency table for the income per person based on a number of ranges (arbitrarily chosen are):

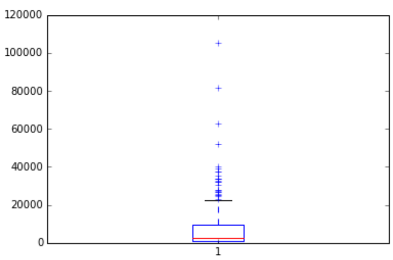
Low Income,1000-5000 61
Very Low Income,0-1000 54
Okay Income,5000-10000 28
Great Income,20000-50000 26
Good Income,10000-20000 17
50,000-200,000 4

When comparing the global tot he North Western European countries, the frequencies are totally different, (this is done by selecting rows just for Belgium,France,Netherlands,Ireland,United Kingdom,Germany,Denmark,Sweden,Norway,Finland):

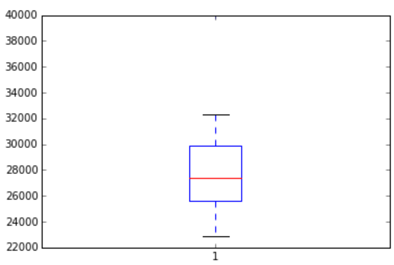
Great Income,20000-50000 10
50,000-200,000 0
Good Income,10000-20000 0
Okay Income,5000-10000 0
Low Income,1000-5000 0
Very Low Income,0-1000 0

Now lets examine the boxplots for both

Boxplot of global income rates



Boxplot of North West European countries.



Within North Western Europe selection you can see a higher mean compared to the global and also a much tighter range of values.

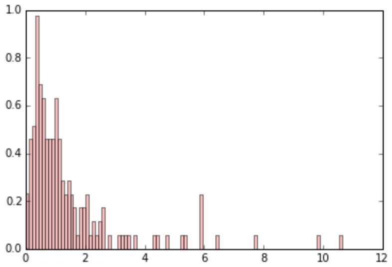
Armed forces rate

Distribution plot of armed forces rate globally (it does not make sense to show this in tabular form)

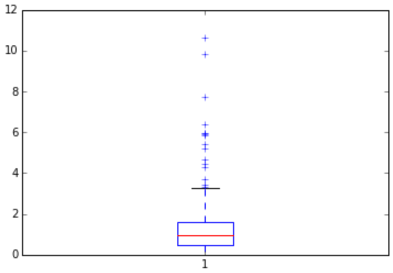
plt.hist(armedforcesrate, bins, normed=True, color="#F08080", alpha=.5);

##Peter Brennan
##13/11/2015
##adding extra categories based upon country
##we will do this by creating functions
“'
five different categories are created
two of these are created using functions
1 by merging an additional dataframe
and two by comparing to lists

”'
##european countries
“'
##european countries
Albania
Andorra
Armenia
Austria
Azerbaijan
Belarus
Belgium
Bosnia
Bulgaria
Croatia
Cyprus
Czech Republic
Denmark
Estonia
Finland
France
Georgia
Germany
Greece
Hungary
Iceland
Ireland
Italy
Kazakhstan
Kosovo
Latvia
Liechtenstein
Lithuania
Luxembourg
Macedonia
Malta
Moldova
Monaco
Montenegro
Netherlands
Norway
Poland
Portugal
Romania
Russia
San Marino
Serbia
Slovakia
Slovenia
Spain
Sweden
Switzerland
Turkey



It is shown as a boxplot below:



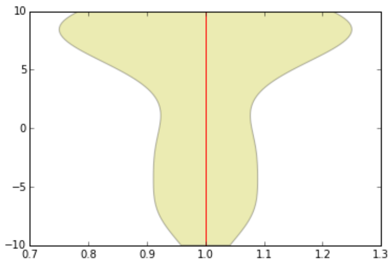
The frequency table is shown in the appendix.

Politly score

Frequency table of Politly scores:

frequency table of politly scores
NaN 0.244131
0 0.028169
1 0.014085
2 0.014085
3 0.009390
4 0.018779
5 0.032864
6 0.046948
7 0.061033
8 0.089202
9 0.070423
10 0.154930
-1 0.018779
-10 0.009390
-9 0.018779
-8 0.009390
-7 0.056338
-6 0.014085
-5 0.009390
-4 0.028169
-3 0.028169
-2 0.023474

Violin Plot of politly scores:



Ukraine
United Kingdom
Vatican City (Holy See) leave this out
https://en.wikipedia.org/wiki/List_of_sovereign_states_and_dependent_territories_in_Europe
#####
'''

```
def EUROPEAN (row):
  if row['country'] == 'Albania' :
    return 'Europe'
  elif row['country'] == 'Andorra' :
    return 'Europe'
  elif row['country'] == 'Armenia' :
    return 'Europe'
  elif row['country'] == 'Azerbaijan' :
    return 'Europe'
  elif row['country'] == 'Austria' :
    return 'Europe'
  elif row['country'] == 'Belarus' :
    return 'Europe'
  elif row['country'] == 'Belgium' :
    return 'Europe'
  elif row['country'] == 'Bosnia' :
    return 'Europe'
  elif row['country'] == 'Bulgaria' :
    return 'Europe'
  elif row['country'] == 'Croatia' :
    return 'Europe'
  elif row['country'] == 'Cyprus' :
    return 'Europe'
  elif row['country'] == 'Czech Republic'
:
    return 'Europe'
  elif row['country'] == 'Denmark' :
    return 'Europe'
  elif row['country'] == 'Estonia' :
    return 'Europe'
  elif row['country'] == 'Finland' :
    return 'Europe'
  elif row['country'] == 'France' :
    return 'Europe'
  elif row['country'] == 'Georgia' :
    return 'Europe'
  elif row['country'] == 'Germany' :
    return 'Europe'
  elif row['country'] == 'Greece' :
    return 'Europe'
  elif row['country'] == 'Hungary' :
    return 'Europe'
  elif row['country'] == 'Iceland' :
    return 'Europe'
  elif row['country'] == 'Ireland' :
    return 'Europe'
  elif row['country'] == 'Italy' :
    return 'Europe'
  elif row['country'] == 'Kazakhstan' :
    return 'Europe'
  elif row['country'] == 'Kosovo' :
    return 'Europe'
  elif row['country'] == 'Latvia' :
    return 'Europe'
  elif row['country'] == 'Liechtenstein' :
    return 'Europe'
  elif row['country'] == 'Lithuania' :
    return 'Europe'
  elif row['country'] == 'Luxembourg' :
```

```
    return 'Europe'
  elif row['country'] == 'Macedonia' :
    return 'Europe'
  elif row['country'] == 'Malta' :
    return 'Europe'
  elif row['country'] == 'Moldova' :
    return 'Europe'
  elif row['country'] == 'Monaco' :
    return 'Europe'
  elif row['country'] == 'Montenegro' :
    return 'Europe'
  elif row['country'] == 'Netherlands' :
    return 'Europe'
  elif row['country'] == 'Norway' :
    return 'Europe'
  elif row['country'] == 'Poland' :
    return 'Europe'
  elif row['country'] == 'Portugal' :
    return 'Europe'
  elif row['country'] == 'Romania' :
    return 'Europe'
  elif row['country'] == 'Russia' :
    return 'Europe'
  elif row['country'] == 'San Marino' :
    return 'Europe'
  elif row['country'] == 'Serbia' :
    return 'Europe'
  elif row['country'] == 'Slovak REpublic'
:
    return 'Europe'
  elif row['country'] == 'Slovenia' :
    return 'Europe'
  elif row['country'] == 'Spain' :
    return 'Europe'
  elif row['country'] == 'Sweden' :
    return 'Europe'
  elif row['country'] == 'Switzerland' :
    return 'Europe'
  elif row['country'] == 'Turkey' :
    return 'Europe'
  elif row['country'] == 'Ukraine' :
    return 'Europe'
  elif row['country'] == 'United Kingdom'
:
    return 'Europe'
  else :
   return 'Not-In-Europe'

data['European'] = data.apply (lambda
row: EUROPEAN (row),axis=1)

##check it worked

"'
Out[24]:
Europe         45
Not-In-Europe    168
dtype: int64
"'

data['European'].value_counts(sort=Fal
se, dropna=False)

"'
##EU countries list as of 2015
Austria,
Belgium,
Bulgaria,
Croatia,
```

```
Cyprus,
Czech Republic,
Denmark,
Estonia,
Finland,
France,
Germany,
Greece,
Hungary,
Ireland,
Italy,
Latvia,
Lithuania,
Luxembourg,
Malta,
Netherlands,
Poland,
Portugal,
Romania,
Slovak Republic,
Slovenia,
Spain,
Sweden,
United Kingdom
## source https://www.gov.uk/eu-eea
##
”’

def EUMEMBER (row):
  if row['country’] == 'Austria’ :
    return 'EU’
  elif row['country’] == 'Belgium’ :
    return 'EU’
  elif row['country’] == 'Bulgaria’ :
    return 'EU’
  elif row['country’] == 'Croatia’ :
    return 'EU’
  elif row['country’] == 'Cyprus’ :
    return 'EU’
  elif row['country’] == 'Czech Republic’
:
    return 'EU’
  elif row['country’] == 'Denmark’ :
    return 'EU’
  elif row['country’] == 'Estonia’ :
    return 'EU’
  elif row['country’] == 'Finland’ :
    return 'EU’
  elif row['country’] == 'France’ :
    return 'EU’
  elif row['country’] == 'Germany’ :
    return 'EU’
  elif row['country’] == 'Greece’ :
    return 'EU’
  elif row['country’] == 'Hungary’ :
    return 'EU’
  elif row['country’] == 'Ireland’ :
    return 'EU’
  elif row['country’] == 'Italy’ :
    return 'EU’
  elif row['country’] == 'Latvia’ :
     return 'EU’
  elif row['country’] == 'Lithuania’ :
     return 'EU’
  elif row['country’] == 'Luxembourg’ :
     return 'EU’
  elif row['country’] == 'Malta’ :
     return 'EU’
```

```
  elif row['country'] == 'Netherlands' :
     return 'EU'
  elif row['country'] == 'Poland' :
     return 'EU'
  elif row['country'] == 'Portugal' :
     return 'EU'
  elif row['country'] == 'Romania' :
     return 'EU'
  elif row['country'] == 'Slovak Republic'
:
     return 'EU'
  elif row['country'] == 'Slovenia' :
     return 'EU'
  elif row['country'] == 'Spain' :
     return 'EU'
  elif row['country'] == 'Sweden' :
     return 'EU'
  elif row['country'] == 'United Kingdom'
:
     return 'EU'
  else :
    return 'Not-In-EU'

data['Eu_Member'] = data.apply
(lambda row: EUMEMBER
(row),axis=1)
##

###NATO mebers and since when they
joined
“'
```

Albania
2009

Belgium
1949

Bulgaria
2004

Canada
1949

Croatia
2009

Czech Republic
1999

Denmark
1949

Estonia
2004

France
1949

Germany
1955

Greece
1952

Hungary
1999

Iceland
1949

Italy
1949

Latvia
2004

Lithuania
2004

Luxembourg
1949

Netherlands
1949

Norway
1949

Poland
1999

Portugal
1949

Romania
2004

Slovak Republic
2004

Slovenia
2004

Spain
1982

Turkey
1952

United Kingdom
1949

United States
1949
###http://www.nato.int/cps/en/natohq/topics_52044.htm
”“

##NATO data

Nato_Countries = pandas.DataFrame({
'country' :
('Albania','Belgium','Bulgaria','Canada','Croatia','Czech Republic','Denmark','Estonia','France','Germany','Greece','Hungary','Iceland','Italy','Latvia','Lithuania','Luxembourg','Netherlands','Norway','Poland','Portugal','Romania','Slovak Republic','Slovenia','Spain','Turkey','United Kingdom','United States'),
'Year_Joined' :
(2009,1949,2004,1949,2009,1999,1949,2004,1949,1955,1952,1999,1949,1949,2004,2004,1949,1949,1949,1999,1949,2004,2004,2004,1982,1952,1949,1949),
'Is_Nato_Country' :
'Nato_Member'
})

##Enhanced data join NATO data

data=pandas.merge(data, Nato_Countries,how='left',on='country')

```
##data.columns.values
##check that all column values have
been added

data.columns.values
##year joined needs to be renamed
data.rename(columns={'Year_Joined':
'Year_Joined_Nato'}, inplace=True)
## change columns names
data.columns.values
##changes are in place

##calculate the age of countries in
NATO
## if not in nato set a decode to set age
to -1

import time
##check how to calc time
print (time.strftime("%Y"))
##write unction to calculate the  age of
NATO countries based on the current
date
def AGE_YEARS (row):
  current_year=time.strftime("%Y")
  if row['Year_Joined_Nato'] >0 :
     return (int(current_year)-
int(row['Year_Joined_Nato']))
  else :
     return -1

##calculate the age of NATO countries
data['Years_In_Nato'] = data.apply
(lambda row: AGE_YEARS
(row),axis=1)
##calculate the age of NAto countries
print("distribution of years in NATO for
Nato Countries")
pp2=data['Years_In_Nato'].value_count
s(sort=False, dropna=False)
print(pp2)
##Mean number of years in NATO by
european or Non EU
print("Mean years of countries in NATO
for European and Non European
Countries")
pp3=data[data['Years_In_Nato']>0]
['Years_In_Nato'].groupby(data['Eu_Me
mber']).mean()
print(pp3)
##Count of countries in EU who are not
in not in NATO
print("Count of countries in NATO for
European and Non European
Countries")
pp4=data[data['Years_In_Nato']>0]
['Years_In_Nato'].groupby(data['Eu_Me
mber']).count()
print(pp4)

##EU 'Eu_Member' Is_Nato_Country
'Nato_Member'

##function tocalculate whether a
country is in both the EU and NATO
def EU_NATO
(Nato_Membership,EU_Membership):
  if Nato_Membership ==
'Nato_Member' and EU_Membership==
'EU' :
```

```
    return 'Nato_And_EU’
  elif Nato_Membership ==
'Nato_Member’ and EU_Membership
== 'Not-In-EU’:
    return 'Nato_Not_In_EU’
  elif Nato_Membership !=
'Nato_Member’ and EU_Membership==
'EU’ :
    return 'Not_In_Nato_In_EU’
  else :
    return 'Not_In_Nato_Not_In_EU’

EU_NATO('Nato_Member’,'EU’)
##test
##apply the function
data['NATO_EU_MEMBERSHIP’] =
data.apply (lambda row:
EU_NATO(row['Is_Nato_Country’],row[
'Eu_Member’]),axis=1)

print(“Nato and EU membership
contingency tables”)
pp5=data['NATO_EU_MEMBERSHIP’].
value_counts(sort=False,
dropna=False)
print(pp5)
##politlyscores for EU countries
print(“distribution of politly score for EU
countries”)
pp6=data[data['Eu_Member’]=='EU’]
['polityscore’].value_counts(sort=True,
dropna=False)
print(pp6)

data.dtypes
##check the object type
##cast to a string
data['NATO_EU_MEMBERSHIP’]=data
['NATO_EU_MEMBERSHIP’].astype(st
r)

##ignore commented out code below
“’
ignore only plotting 3 of 4 categoies
ggplot(data, aes(x='armedforcesrate’,
y='incomeperperson’,
color="NATO_EU_MEMBERSHIP”)) +\
  geom_point() +\
  scale_color_brewer(type='diverging’,
palette=4) +\
  xlab(“armed forces rate”) +
ylab(“IncomePerPerson”) +
ggtitle(“Gapminder”)

####
“’

p=ggplot(data,
aes(x='armedforcesrate’,
y='incomeperperson’))
p + geom_point(alpha=0.25) + \

 facet_grid("NATO_EU_MEMBERSHIP
”)

##    subset for europe
data.columns.values
subset_data_europe=data[data['Europe
an’]=='Europe’]
```

```
print(“Nato and EU membership contingency tables for europe only”)
epp5=subset_data_europe['NATO_EU_MEMBERSHIP’].value_counts(sort=False, dropna=False)
print(epp5)

p=ggplot(subset_data_europe, aes(x='armedforcesrate’, y='incomeperperson’))
p + geom_point(alpha=0.25) + \

 facet_grid(“NATO_EU_MEMBERSHIP”)
```

Show more