PORTFOLIO DATA SCIENCE BOOTCAMP

Priscila Boada

NORTHUMBRIA UNIVERSITY NEWSCASTLE

Contents

Data Analysis 3
Task 1 3
Task 1.1 Type of data 3
Task 1.2 Error correction method 3
Task 2 Evidence your histograms 4
Task 3 4
Task 3.1 Evidence your new vector 5
Task 3.2 Evidence your scatter plot 6
Task 3.3 document your new variable 7
Task 4 Features with missing data 8
Task 5 Method for replacing missing values 9
Task 6 Document your new feature 17
Task 7 Document the size of induration 18
Exploratory Data Analysis 18
Task 1 Evidence table of findings 18
Task 2 Provide correlation chart 20
Task 3 Document how techniques can help to satisfy assumptions 21
Classification and Clustering 22
Task 1 Predictive accuracy via a Naïve Bayes model 22
Task 2 Evidence the random forest table which identifies best performance 23
Task 3 Evidence the kernel table which identifies best performance 24
Task 4 Validate best method 24
Task 5 Provide table 24

Task 6 Record the parameters that were used for the Random Forest Regressor 25
Task 6.1 What does it tell you? 25
Task 6.2 How does it compare? ..... **Error! Bookmark not defined.**

Data Analysis

Task 1

Using the iris dataset from week 4, load the iris data and convert the data to a dataframe.

Task 1.1 Type of data

Find out the structure of this dataframe. What data type has been used for Petal.Length?

It has been used a float64 as data type for petallength

Evidence

```
iris= pd.read_csv('D:/respaldo/Data Science Northumbria/python_Northumbria/iris.csv')
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   sepallength  150 non-null    float64
 1   sepalwidth   150 non-null    float64
 2   petallength  150 non-null    float64
 3   petalwidth   150 non-null    float64
 4   class        150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

Task 1.2 Error correction method

You have received notice that all of the plants marked as "setosa" have been incorrectly labelled and should instead be recorded as "junos". Update the data to reflect this. Describe how you went about updating the plant label to correct the error in the data surrounding *Setosa* to *Junos*

To replace the value in the data Frame, we select the column class, which holds the description 'Iris-setosa' and then we replace this description by Junos

Evidence:

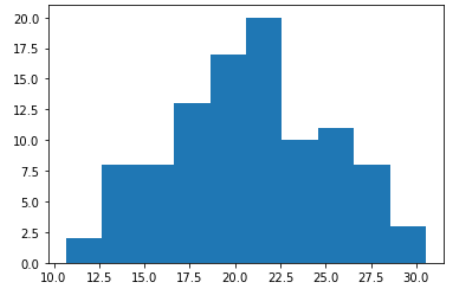
```
In [32]: iris['class']=iris['class'].replace(['Iris-setosa'],'junos')
         iris[:25]
```

Task 2 Evidence your histograms

Create a dataset of 100 observations with 3 features. The first should follow a normal distribution, with a mean of 20 and a standard deviation of 4. The second feature should follow uniform distribution, with values between 15 and 25, and the third has a poisson distribution with a lambda value of 5. Produce a series of histograms showing the distribution of each variable.

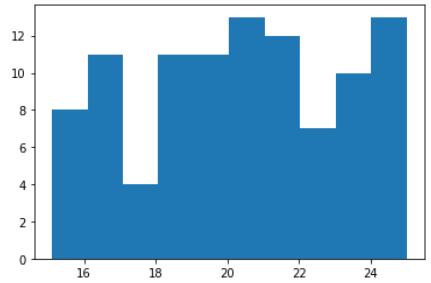
Normal distribution

```
normalDistribution = np.random.normal(20,4,100)
plt.hist(normalDistribution)
plt.show()
```



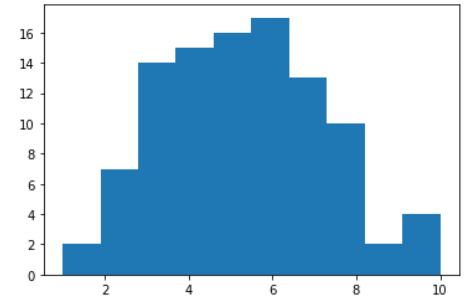
Uniform Distribution

```
uniformDistribution = np.random.uniform(15,25,100)
plt.hist(uniformDistribution)
plt.show()
```



Poisson Distribution

```
PoissonDistribution = np.random.poisson(5,100)
plt.hist(PoissonDistribution)
plt.show()
```



Task 3

Using the mtcars data from week 4, load mtcars. An approximation of the 0-60 time of a car can be calculated through the formula:

$$\left({}^{1}/_{(Horsepower\ /\ Weight)}\right) \cdot 440$$

Task 3.1 Evidence your new vector

Using this, create a new vector of 0-60 times for the cars, and then attach this to the dataset. Evidence your code for creating a new vector of 0-60 times

```
zero_sixty=1/(mtcars['hp']/mtcars['wt'])*440
mtcars['zero to sixty']=zero_sixty
mtcars
```

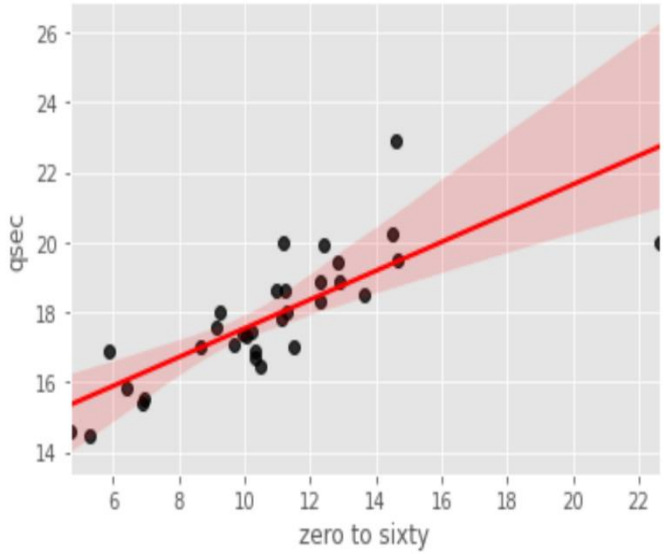
	name	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	zero to sixty
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4	10.480000
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4	11.500000
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1	10.976344
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1	12.860000
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2	8.649143
5	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1	14.499048
6	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4	6.411429
7	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2	22.638710
8	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2	14.589474
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4	12.305691
10	Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4	12.305691
11	Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3	9.948889
12	Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3	9.117778
13	Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3	9.240000
14	Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4	11.268293
15	Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4	11.100279
16	Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4	10.225217
17	Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1	14.666667
18	Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2	13.665385
19	Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1	12.421538
20	Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1	11.181443
21	Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2	10.325333
22	AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2	10.076000
23	Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4	6.896327
24	Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	0	3	2	9.667429
25	Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1	12.900000
26	Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2	10.347253
27	Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2	5.891327
28	Ford Panters L	15.8	8	351.0	264	4.22	3.170	14.50	0	1	5	4	5.283333
29	Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6	6.964571
30	Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	1	5	8	4.688955
31	Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2	11.222018

Task 3.2 Evidence your scatter plot

Create a scatter plot showing the relationship between the 0-60 time and the quarter-mile time. Illustrate this relationship with a linear regression line, coloured red. What does this tell us about the relationship between these features? Paste evidence of the scatter plot showing the relationship between the 0-60 time and the quarter-mile time.

```
import pandas as pd
import seaborn as sns
sns.regplot(x=mtcars['zero to sixty'],y=mtcars['qsec'],scatter_kws={"color": "black"}, line_kws={"color": "red"})
```

```
<AxesSubplot:xlabel='zero to sixty', ylabel='qsec'>
```



As the graphic shows the the quarter-mile time (qsec) and 0-60 time are positively associated, therefore a higher 0-60 time implies a higher quarter-mile time too.

Task 3.3 document your new variable

Create a new variable showing whether a car is classed as "fast" or "slow". A fast car has a 0-60 of less than 7 seconds, otherwise it is classed as slow. Provide evidence of the code you created.

```
mtcars['Class']=np.where(mtcars['zero to sixty']<=7,'fast','slow')
mtcars.head(15)
```

	name	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	zero to sixty	Class
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4	10.480000	slow
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4	11.500000	slow
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1	10.976344	slow
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1	12.860000	slow
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2	8.649143	slow
5	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1	14.499048	slow
6	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4	6.411429	fast
7	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2	22.638710	slow
8	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2	14.589474	slow
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4	12.305691	slow
10	Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4	12.305691	slow
11	Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3	9.948889	slow
12	Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3	9.117778	slow
13	Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3	9.240000	slow
14	Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4	11.268293	slow

Task 4 Features with missing data

Document your answers in relation to which features within the immunotherapy.csv dataset

a. What features within the dataset have missing data?

The columns below have missing data

```
df.isna().sum()
```

```
Unnamed: 0              0
sex                     0
age                     4
Time                    0
Number_of_Warts         7
Type                    3
Area                    0
induration_diameter     0
Result_of_Treatment     0
dtype: int64
```

b. What percentage of the total dataset is missing. What percentage of data is missing for the features identified in Part A.

```
# percentage of the total dataset missing
missing=sum(df['age'].isna()==True)+sum(df['Type'].isna()==True)+sum(df['Number_of_Warts'].isna()==True)
missingness=missing/df.shape[0]*100
print('There are',missing, 'missing data which is the','%.2f'%+missingness, '% of the dataset')
```

```
There are 14 missing data which is the 15.56 % of the dataset
```

```
#percentage of missing data per feature
df.isna().sum()/df.shape[0]*100
```

```
Unnamed: 0            0.000000
sex                   0.000000
age                   4.444444
Time                  0.000000
Number_of_Warts       7.777778
Type                  3.333333
Area                  0.000000
induration_diameter   0.000000
Result_of_Treatment   0.000000
dtype: float64
```

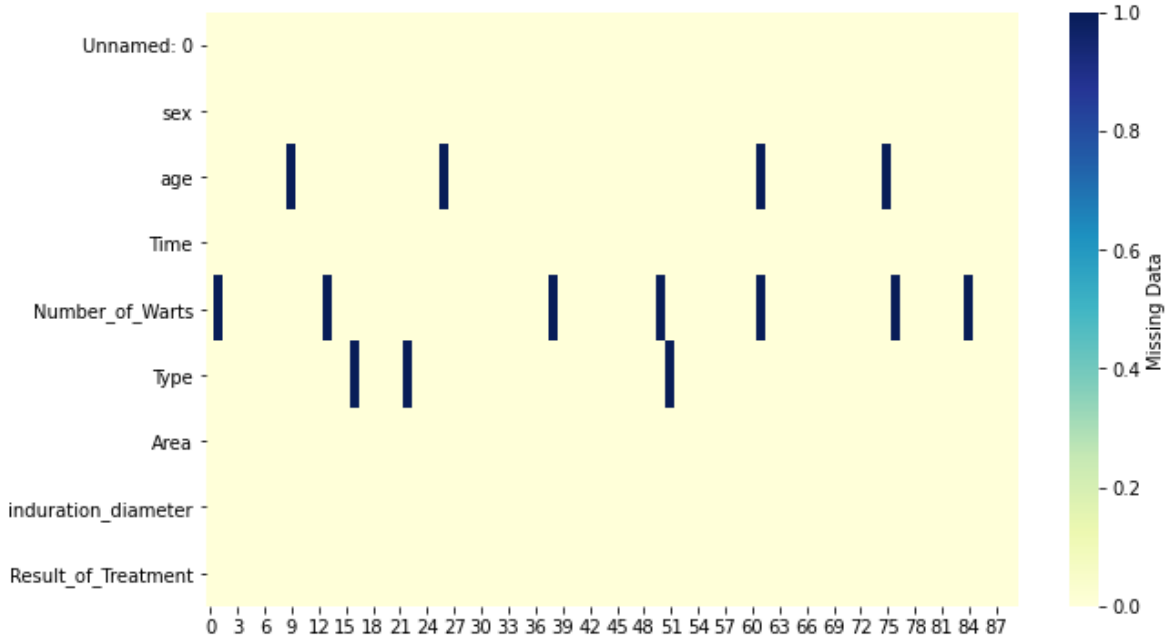
Task 5 Method for replacing missing values

Describe the chosen method for replacing missing values in each of the features and evidence your use of this method to replace the values so that each feature has complete data.

Method

1. Apply the little test to find out if the data is missing completely at ramdom

```
#Little's test
import seaborn as sns
from matplotlib import pyplot as plt
plt.figure(figsize=(10,6))
sns.heatmap(df.isna().transpose(),
    cmap="YlGnBu",
    cbar_kws={'label': 'Missing Data'})
plt.show()
```

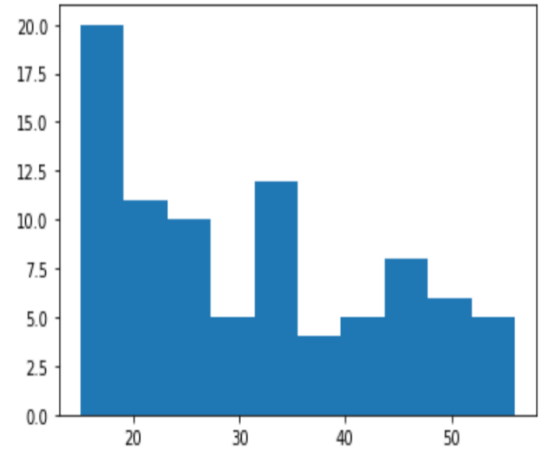


the heatmap shows that the values have a p.value>0.05 therefore the data can be treated as MCAR

2. Treating data per column: Analize the distribution, find out what method of imputation can be use and replace the values

Age

```
plt.hist(df['age'])
plt.show()
```

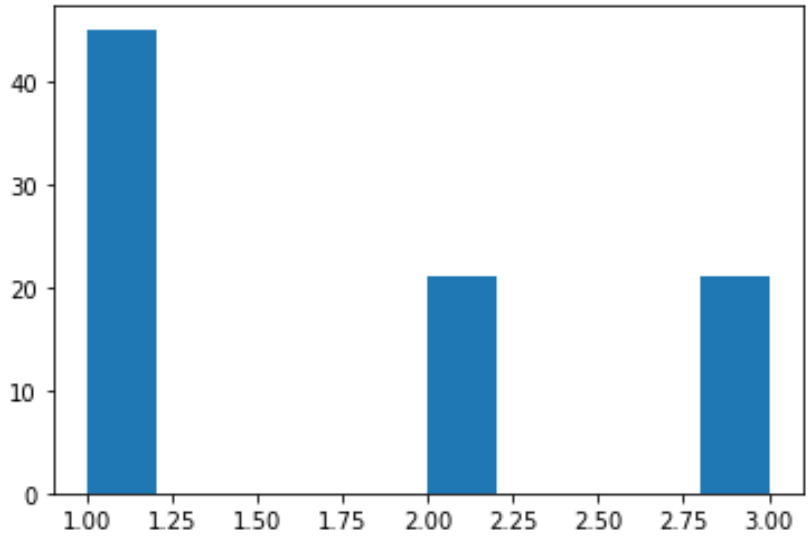



```
#Since it is a skewed distributions, the median is better than the mean because it is not influenced
# by extremely large values
import numpy as np
median = df['age'].median()
df['age'].replace(np.NAN,median,inplace=True)
df.head(20)
```

	Unnamed: 0	sex	age	Time	Number_of_Warts	Type	Area	induration_diameter	Result_of_Treatment
0	1	1	22.0	2.25	14.0	3.0	51	50	1
1	2	1	15.0	3.00	NaN	3.0	900	70	1
2	3	1	16.0	10.50	2.0	1.0	100	25	1
3	4	1	27.0	4.50	9.0	3.0	80	30	1
4	5	1	20.0	8.00	6.0	1.0	45	8	1
5	6	1	15.0	5.00	3.0	3.0	84	7	1
6	7	1	35.0	9.75	2.0	2.0	8	6	1
7	8	2	28.0	7.50	4.0	1.0	9	2	1
8	9	2	19.0	6.00	2.0	1.0	225	8	1
9	10	2	28.5	12.00	6.0	3.0	35	5	0
10	11	2	33.0	6.25	2.0	1.0	30	3	1
11	12	2	17.0	5.75	12.0	3.0	25	7	1
12	13	2	15.0	1.75	1.0	2.0	49	7	0
13	14	2	15.0	5.50	NaN	1.0	48	7	1
14	15	2	16.0	10.00	7.0	1.0	143	6	1
15	16	2	33.0	9.25	2.0	2.0	150	8	1
16	17	2	26.0	7.75	6.0	NaN	6	5	1
17	18	2	23.0	7.50	10.0	2.0	43	3	1
18	19	2	15.0	6.50	19.0	1.0	56	7	1
19	20	2	26.0	6.75	2.0	1.0	6	6	1

Type

In [34]:
```
plt.hist(df['Type'])
plt.show()
```



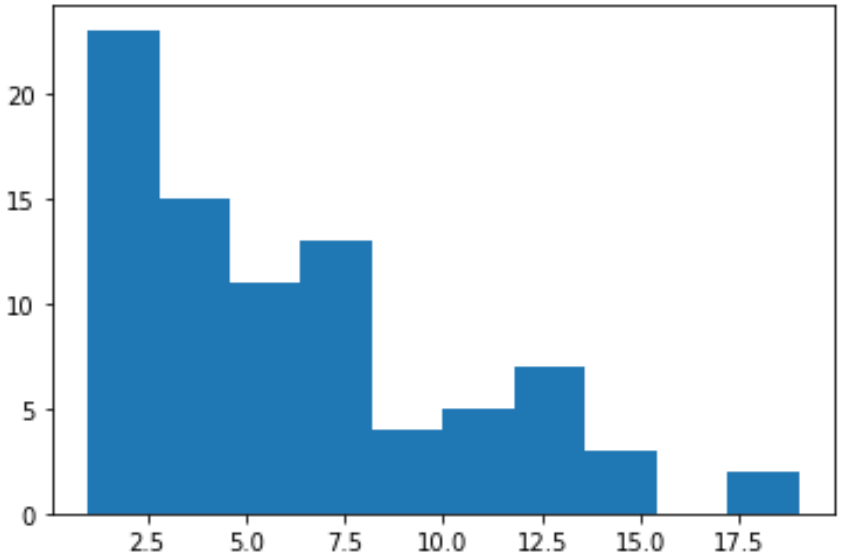
In [46]:
```
#For this distribution, since there are repetitive values, it will be
#taken the mode as the most representative value
import numpy as np
import statistics
mode = statistics.mode(df['Type'])
df['Type'].replace(np.NAN,mode,inplace=True)
df.head(20)
```

Out[46]:

	Unnamed: 0	sex	age	Time	Number_of_Warts	Type	Area	induration_diameter	Result_of_Treatment
0	1	1	22.0	2.25	14.0	3.0	51	50	1
1	2	1	15.0	3.00	NaN	3.0	900	70	1
2	3	1	16.0	10.50	2.0	1.0	100	25	1
3	4	1	27.0	4.50	9.0	3.0	80	30	1
4	5	1	20.0	8.00	6.0	1.0	45	8	1
5	6	1	15.0	5.00	3.0	3.0	84	7	1
6	7	1	35.0	9.75	2.0	2.0	8	6	1
7	8	2	28.0	7.50	4.0	1.0	9	2	1
8	9	2	19.0	6.00	2.0	1.0	225	8	1
9	10	2	28.5	12.00	6.0	3.0	35	5	0
10	11	2	33.0	6.25	2.0	1.0	30	3	1
11	12	2	17.0	5.75	12.0	3.0	25	7	1
12	13	2	15.0	1.75	1.0	2.0	49	7	0
13	14	2	15.0	5.50	NaN	1.0	48	7	1
14	15	2	16.0	10.00	7.0	1.0	143	6	1
15	16	2	33.0	9.25	2.0	2.0	150	8	1
16	17	2	26.0	7.75	6.0	1.0	6	5	1
17	18	2	23.0	7.50	10.0	2.0	43	3	1
18	19	2	15.0	6.50	19.0	1.0	56	7	1
19	20	2	26.0	6.75	2.0	1.0	6	6	1

Number of Warts

```
plt.hist(df['Number_of_Warts'])
plt.show()
```




```
#Since it is a skewed distributions, the median is better than the mean because
#it is less influenced by outliers
import numpy as np
median_Warts = df['Number_of_Warts'].median()
df['Number_of_Warts'].replace(np.NAN,median_Warts,inplace=True)
df.head(20)
```

	Unnamed: 0	sex	age	Time	Number_of_Warts	Type	Area	induration_diameter	Result_of_Treatment
0	1	1	22.0	2.25	14.0	3.0	51	50	1
1	2	1	15.0	3.00	6.0	3.0	900	70	1
2	3	1	16.0	10.50	2.0	1.0	100	25	1
3	4	1	27.0	4.50	9.0	3.0	80	30	1
4	5	1	20.0	8.00	6.0	1.0	45	8	1
5	6	1	15.0	5.00	3.0	3.0	84	7	1
6	7	1	35.0	9.75	2.0	2.0	8	6	1
7	8	2	28.0	7.50	4.0	1.0	9	2	1
8	9	2	19.0	6.00	2.0	1.0	225	8	1
9	10	2	28.5	12.00	6.0	3.0	35	5	0
10	11	2	33.0	6.25	2.0	1.0	30	3	1
11	12	2	17.0	5.75	12.0	3.0	25	7	1
12	13	2	15.0	1.75	1.0	2.0	49	7	0
13	14	2	15.0	5.50	6.0	1.0	48	7	1
14	15	2	16.0	10.00	7.0	1.0	143	6	1
15	16	2	33.0	9.25	2.0	2.0	150	8	1
16	17	2	26.0	7.75	6.0	1.0	6	5	1
17	18	2	23.0	7.50	10.0	2.0	43	3	1
18	19	2	15.0	6.50	19.0	1.0	56	7	1
19	20	2	26.0	6.75	2.0	1.0	6	6	1

Task 6 Document your new feature

To make the identification of potentially troublesome issues for patients, you have been requested to create a new feature recording the induration diameter in a more straightforward way

Document your code for computing a new feature, with the induration diameter coded as Small, Medium and Large.

- Small, when the diameter is less than 20.
- Medium, when the diameter ranges from 20 to 50.
- Large, when the diameter is greater than 50.

```
df['induration_diameter_class']=np.where(df.induration_diameter<20,"Small",
                                         np.where(df.induration_diameter<50,"Medium","Large"))
df.head(10)
```

:

	Unnamed: 0	sex	age	Time	Number_of_Warts	Type	Area	induration_diameter	Result_of_Treatment	induration_diameter_class
0	1	1	22.0	2.25	14.0	3.0	51	50	1	Large
1	2	1	15.0	3.00	6.0	3.0	900	70	1	Large
2	3	1	16.0	10.50	2.0	1.0	100	25	1	Medium
3	4	1	27.0	4.50	9.0	3.0	80	30	1	Medium
4	5	1	20.0	8.00	6.0	1.0	45	8	1	Small
5	6	1	15.0	5.00	3.0	3.0	84	7	1	Small
6	7	1	35.0	9.75	2.0	2.0	8	6	1	Small
7	8	2	28.0	7.50	4.0	1.0	9	2	1	Small
8	9	2	19.0	6.00	2.0	1.0	225	8	1	Small
9	10	2	28.5	12.00	6.0	3.0	35	5	0	Small

Task 7 Document the size of induration

Based upon this new representation of the data, which size of induration appears most frequently?

The most frequent value is "small" with 69 appearances.

```
df['induration_diameter_class'].value_counts(ascending=True)
```

```
Large      9
Medium    12
Small     69
Name: induration_diameter_class, dtype: int64
```

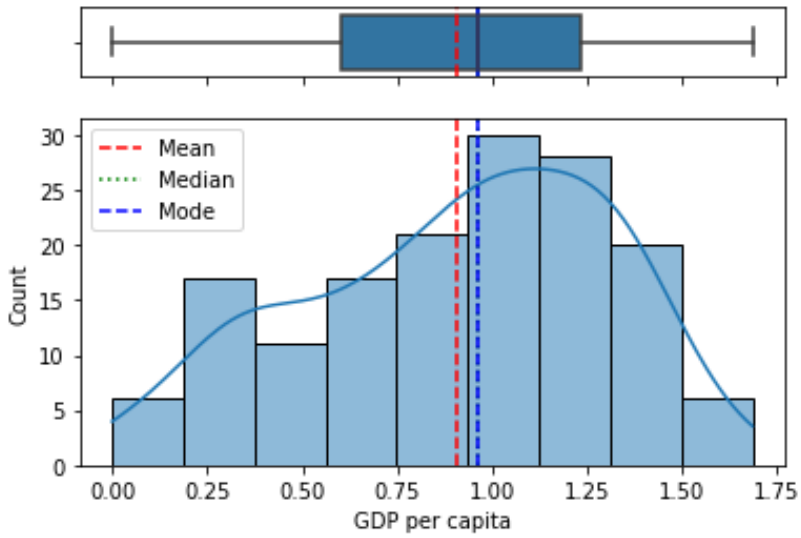
Exploratory Data Analysis

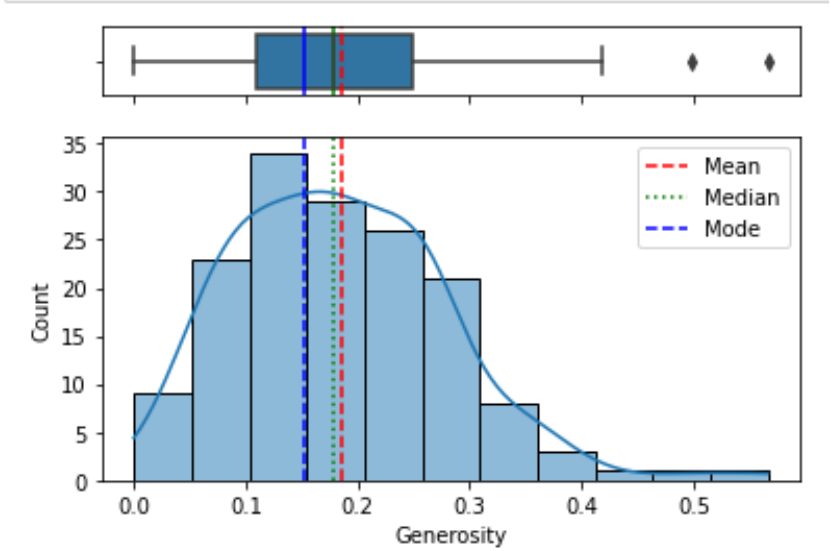
For the tasks below, happiness.csv is provided, and load it as Pandas dataframe. Remove the feature containing the country names.

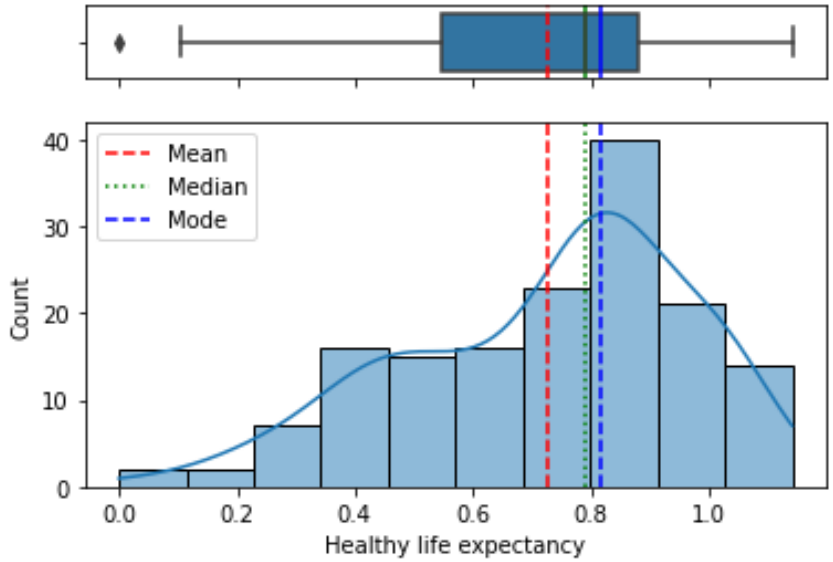
Task 1 Evidence table of findings

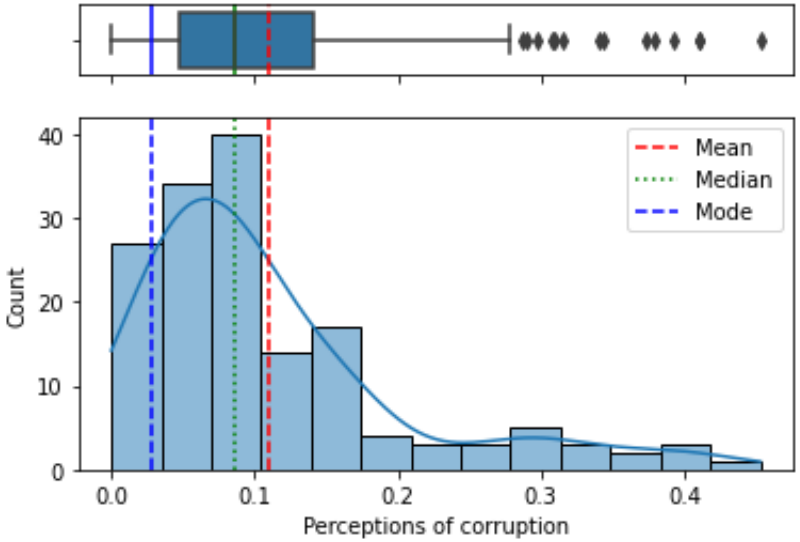
Perform some univariate EDA on the features within the dataset. Complete the table below

Feature	Skewness	Kurtosis	Appropriate measure of central tendency MoCT	MoCT Value	Outliers?
GDP per capita	Negative (-0.385)	Negative (-0.770)	Median	0.960	No
Generosity	Positive (0.746)	Positive (1.173)	Median	0.178	Yes
Healthy life expectancy	Negative (-0.614)	Negative (-0.303)	Median	0.789	Yes
Perceptions of corruption	Positive (1.650)	Neutral (2.412*) *Value close to 3	Median	0.086	Yes


Mean
Median
Mode
Count
GDP per capita

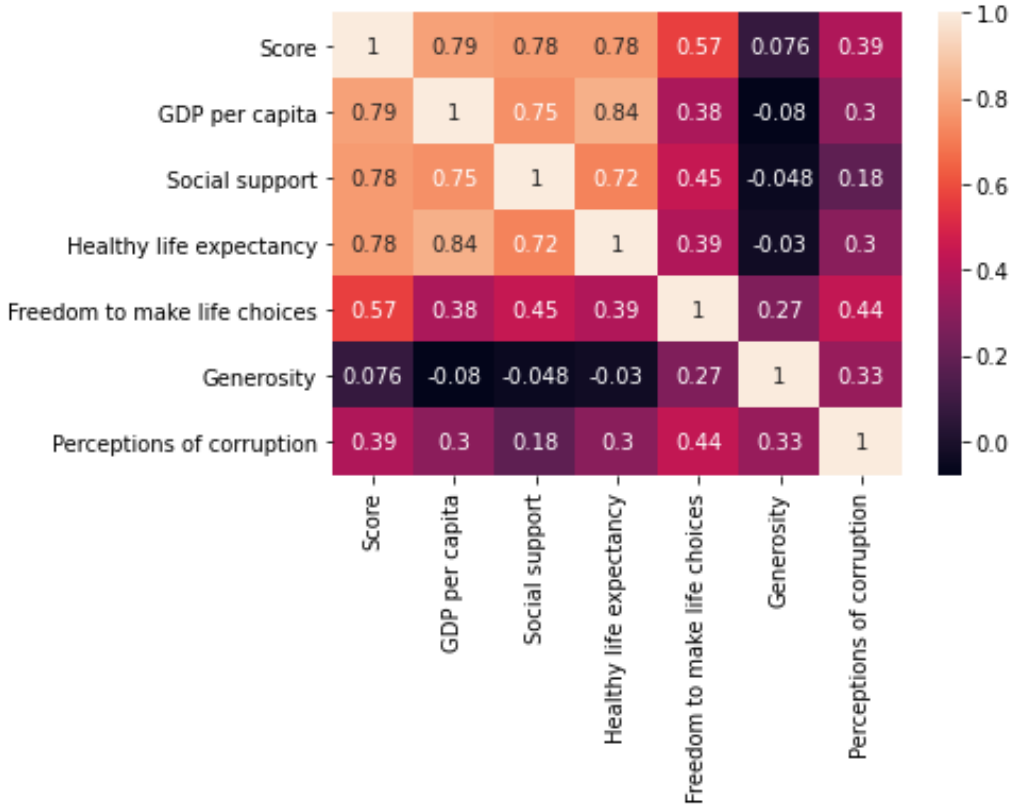

Mean
Median
Mode
Count
Generosity


Mean
Median
Mode
Count
Healthy life expectancy


Mean
Median
Mode
Count
Perceptions of corruption

Task 2 Provide correlation chart

Produce a correlation chart for the dataset. When developing a model to predict the overall satisfaction score:



2.1. Which features would act as the strongest predictor?

The strongest predictors are the ones that have the strongest correlation, meaning that the Pearson correlation coefficient is greater than $|\pm 0.6|$. The features that have correlations with a score greater than 0.6 are:

- GDP per capita (0. 79)
- Social support (0.78)
- Healthy life expectancy (0.78)

2.2. Which features may you wish to remove from the dataset, and why?

GDP per capita or Healthy life expectancy because these features are strongly correlated (correlation>0.8). Therefore, if both features are used, there will be issues with multicollinearity.

Task 3 Document how techniques can help to satisfy assumptions

3. Document how the techniques covered thus far can help to satisfy:

3.1. The assumption of feature independence

The independence of a feature can be identified using the Pearson Correlation feature. As it is displayed in the chart, values higher than 0.8 may indicate a strong correlation and therefore this could signify that the features are dependent on each other.

The features that have a high degree of interdependency for this dataset are GDP per capita and healthy life expectancy. Thus, by using the Pearson Correlation technique we can identify that the other features can be assumed as independent.

3.2. The assumption of observational independence

There is observational independence when 'any data point in a set of data is statistically independent from the rest.' This means that its value is not influenced by the value of any other observation in the set, thus the observations are not correlated.

A way to identify observational independence is by plotting the distribution of each feature. If is seen variation between the data, then it can be assumed observational independence. On the contrary, if the distribution is skewed, this could signify that the outliers are somehow correlated.

3.3. The assumption of the approximation of normality

The technique that helps to identify how close to an approximation of normality is a quantile-normal plot. By plotting a line that represents a normal distribution and the feature analyzed, it is identified any deviation that shows non-normality.

3.4. The assumption of accurate data

To determine if the data is accurate, the technique to use is to plot the distribution of each feature. If outliers or unusual data are found, and there is no cause for them to exist then it could be thought that the data might have some errors, therefore is not accurate.

Classification and Clustering

The evidence required for these tasks relates to **Week 9** in which you were using a variety of clustering and classification techniques. Note only the work developed in the **Tasks** section of **Chapter 5 – Tasks Classification Models** is required to be evidenced here.

Task 1 Predictive accuracy via a Naïve Bayes model

Try to build and test Naïve Bayes model yourself. Evidence the predictive accuracy you obtained through the creation of the Naïve Bayes model you created for **Task 1**

The accuracy is 0.753

Naïve Bayes model

In [42]:
```
from sklearn.naive_bayes import GaussianNB
```

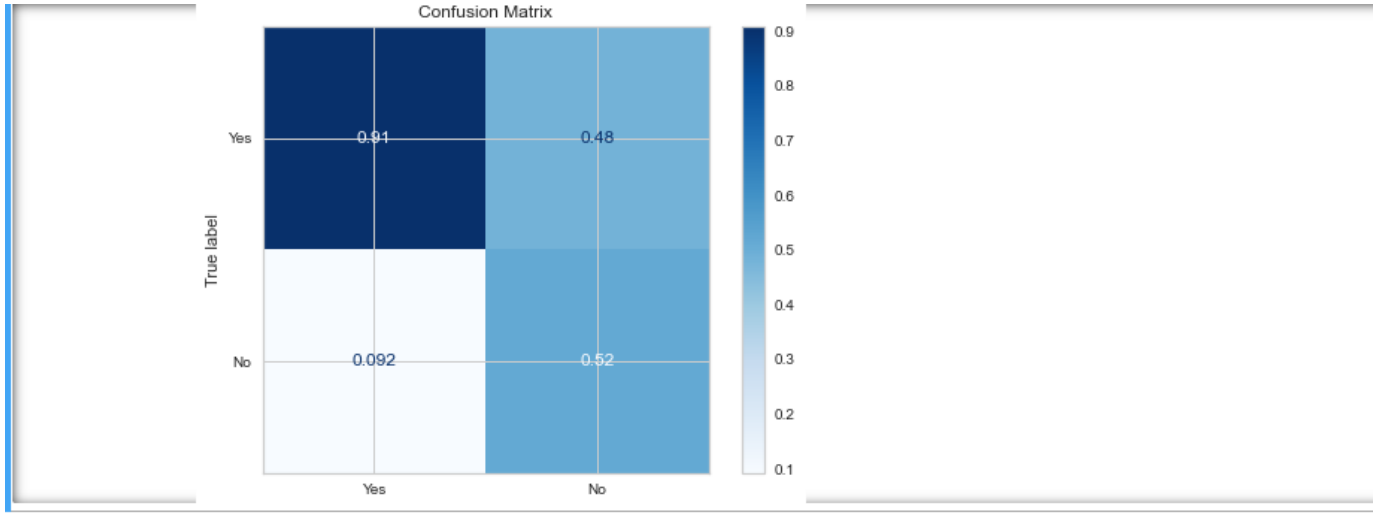
In [44]:
```
gnbModel = GaussianNB()
gnbModel = gnbModel.fit(features,target)
```

```
C:\Users\prisc\AppData\Roaming\Python\Python38\site-packages\sklearn\utils\validation.py:993: DataConversionWarning: A co
lumn-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using r
avel().
```

In [48]:
```
gnbpredicted= gnbModel.predict(testFeatures) #All rows prediction
print(gnbpredicted)
```

```
[0 1 0 ... 0 0 0]
```

In [49]:
```
from sklearn.metrics import plot_confusion_matrix
from sklearn.metrics import ConfusionMatrixDisplay
figure(figsize=(16, 6), dpi=80)
plot_confusion_matrix(gnbModel, testFeatures, testTarget,display_labels=cn,normalize='pred', cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.show()
```




```
In [50]: from sklearn.metrics import accuracy_score
         accScore = accuracy_score(testTarget, gnbpredicted)
         accScore

Out[50]: 0.7526652452025586
```

Task 2 Evidence the random forest table which identifies best performance

The process of tuning the parameters of a model to attempt to maximise the performance is called parameter optimisation. Complete the table below, tuning the parameters for the random forest with values of your choosing and recording the results (the first row represents the example completed in the exercise). Which combination performs the best?

Attempt Number	Max_features	n_estimators	Accuracy%
1	5	500	0.8017057569296375
2	5	1000	0.8024164889836531
3	5	200	0.7974413646055437
4	15	500	0.7967306325515281
5	3	500	0.783226723525231

The best performance corresponds to attempt 2, where Max_features= 5, n_estimators=1000 and the accuracy is 0.802.

Task 3 Evidence the kernel table which identifies best performance

One of the key parameters that can be tuned when creating an SVM model is the choice of kernel that is used. There are three forms of kernel that can be used: rbf, linear, poly and sigmoid. For example, to use a linear kernel, the following command would be used:

SvmModel = svm.SVC(kernel='rbf',gamma='scale')

Complete the table below, along with the example from the exercise. Which kernel performs the best?

Attempt Number	Kernel	Accuracy (%)
1	rbf kernel	0.7093105899076049
2	Linear kernel	0.7938877043354655
3	Polynomial kernel	0.7398720682302772
4	Sigmoid kernel	0.7057569296375267

The kernel that performs the best is the linear kernel having an accuracy of 0.794.

Task 4 Validate best method

Document why you think that method produced the best results in the task above in terms of predictive capacity as per **Task 4.**

The method that produced the best results is Random Forest because its accuracy is slightly higher (0.802) than the SVM accuracy (0.794). Moreover, the performance of the Random Forest algorithm was faster than the SVC model. Therefore, for this data set, the best method is Random Forest.

Task 5 Provide table

Provide the completed table which documents your construction of a multivariate linear model, using "target" as the response variable and three of the continuous features available within the dataset as per **Task 2.**

Model attempt	IF1 NAME	Is significant?	IF2 NAME	Is significant?	IF3 NAME	Is significant?	R-squared	Model accuracy (as correlation coefficient)
1	Age	Yes					0.171	-0.215146
2	Age	Yes	CRIM	Yes			0.241	-0.155104
	Age	Yes	ZN	Yes				
3	Age	Yes	CRIM	Yes	DIS	Yes	0.250	-0.093491
	Age	Yes	CRIM	Yes	INDUS	Yes		
	Age	Yes	RM	Yes	DIS	Yes		

Task 6 Record the parameters that were used for the Random Forest Regressor

Build a Random Forest Regressor using this data. Record the parameters that were used for the RF regressor.

Task 6.1 What does it tell you?

Document the importance of features using Randomized Search, and their standardised beta coefficients as per **Task 3 a)**

Analysing the coefficients obtained for this model, it is clear that the feature that has a higher impact on the prediction of MEDV is Age, then is followed by CRIM and finally DIS. This can be told through the weights of the standardised beta coefficients.

```
Best parameters:  {'model__alpha': 0.1}
Coefficients:  [-3.38046686 -2.5708594  -0.8840058 ]
[3.38046686 2.5708594  0.8840058 ]
```

Task 6.2 How does it compare?

Document how the predictive performance of the model compares with the linear models

the Random Forest model used with the features 'AGE','CRIM','DIS' has a $R^2 = 0.676$. In contrast, the multilinear model for the same features has a $R^2 = 0.250$. Comparing both values, it is concluded that the Random Forest model has better predictive performance since the R^2 is higher than the computed for the multilinear mode