HW1_BasicStatistics

October 17, 2020

1 Homework Assignement 1 - Basic Statistics - revision notebook

Beatriz Gamboa Pereira - 201705220

[41]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import statsmodels.api as sm
```

In this work, I will be exploring the dataset used in the first practical lesson, namely SAheart, from kaggle.

[2]:
```
mydata = pd.read_csv('SAheart.csv')
mydata
```

[2]:
```
     sbp  tobacco    ldl  adiposity  famhist  typea  obesity  alcohol  age chd
0    160    12.00   5.73      23.11  Present     49    25.30    97.20   52  Si
1    144     0.01   4.41      28.61   Absent     55    28.87     2.06   63  Si
2    118     0.08   3.48      32.28  Present     52    29.14     3.81   46  No
3    170     7.50   6.41      38.03  Present     51    31.99    24.26   58  Si
4    134    13.60   3.50      27.78  Present     60    25.99    57.34   49  Si
..   …        …      …          …        …      …        …        …    … ..
457  214     0.40   5.98      31.72   Absent     64    28.45     0.00   58  No
458  182     4.20   4.41      32.10   Absent     52    28.61    18.72   52  Si
459  108     3.00   1.59      15.23   Absent     40    20.09    26.64   55  No
460  118     5.40  11.61      30.79   Absent     64    27.35    23.97   40  No
461  132     0.00   4.82      33.41  Present     62    14.70     0.00   46  Si

[462 rows x 10 columns]
```

In this sample, we can see that we have 462 **data objects** (patients) each with 10 **attributes**. Of those 10 attributes, 8 constitute numeric variables, either integers or floats; 1 is nominal, namely 'famhist'; and then we have 'chd', which is the class indicator of whether the patient has Coronary Heart Disease or not. This last one is the variable we want to predict as we analyse our data. We can test this by checking the data types of our dataset.

[3]:
```
#checking the number of data objects and attributes
mydata.shape
```

[3]: (462, 10)

[4]:
```
#checking the types of the attributes
mydata.dtypes
```

[4]:
```
sbp            int64
tobacco      float64
ldl          float64
adiposity    float64
famhist       object
typea          int64
obesity      float64
alcohol      float64
age            int64
chd           object
dtype: object
```

1.1 Central tendency measures

Central tendency measures tell us where the sample is located, and how skewed it is realtive to a normal distribution. The most common measures of central tendency are the mean, the median and the mode. The **mean** is the sum of all measurements divided by the number of measurements.

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

where N is the number of observations in the sample and $\{x_1, x_2, \ldots, x_N\}$ are the observed values of the sample items. We can also have a **weighted mean** where data points have different contributions to the final value of the mean. When all data point have the same contribution, we get the mean as described before.

$$\bar{x} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}$$

where $\{w_1, w_2, \ldots, w_n\}$ are the weights. The **median** is the middle value of your sample, which splits it into halves with equal amounts of datapoints. The **mode** is simply the most frequent value in the dataset. Therefore, we can have a mode for nominal variables, as well as numerical ones.

1.1.1 Mean

[5]:
```
print(mydata.mean(),'\n\n')

#we can check the mean is the sum of all measurements divided by the number of␣
↪measurements
numeric_columns = mydata.select_dtypes([np.number]).columns
for col in numeric_columns:
    sum = 0
    for i in mydata[col]:
        sum += i
    mean = sum / 462
```

```
    print(col.ljust(10),round(mean,6))

#as we can see, the results are the same
```

```
sbp          138.326840
tobacco        3.635649
ldl            4.740325
adiposity     25.406732
typea         53.103896
obesity       26.044113
alcohol       17.044394
age           42.816017
dtype: float64


sbp        138.32684
tobacco    3.635649
ldl        4.740325
adiposity  25.406732
typea      53.103896
obesity    26.044113
alcohol    17.044394
age        42.816017
```

1.1.2 Median

[6]:
```
print(mydata.median(),'\n\n')

#we can check it's the middle value of each column
#since we have 462 datapoints, the median will be the average of the two middle␣
↪values, 230 and 231
for col in numeric_columns:
    sorteda = np.sort(np.array(mydata[col]))
    median = (sorteda[230] + sorteda[231]) / 2
    print(col.ljust(10),round(median,3))

#as we can see, the results are the same
```

```
sbp          134.000
tobacco        2.000
ldl            4.340
adiposity     26.115
typea         53.000
obesity       25.805
alcohol        7.510
age           45.000
dtype: float64
```

```
sbp          134.0
tobacco      2.0
ldl          4.34
adiposity    26.115
typea        53.0
obesity      25.805
alcohol      7.51
age          45.0
```

1.1.3 Mode

[7]:
```python
mydata.mode()
#where we have more than one value with se same frequency, we have more than␣
↪one mode, hence the 4 rows
```

[7]:

	sbp	tobacco	ldl	adiposity	famhist	typea	obesity	alcohol	age	chd
0	134.0	0.0	3.57	21.10	Absent	52.0	24.86	0.0	16.0	No
1	136.0	NaN	3.95	27.55	NaN	NaN	26.09	NaN	NaN	NaN
2	NaN	NaN	4.37	29.30	NaN	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	30.79	NaN	NaN	NaN	NaN	NaN	NaN

1.2 Dispersion measures

Dispersion measures tell us how far apart the measures are, indicating if the distribution is more squeezed or stretched. We have the **range** of our measurements, which is the difference between the minimum and maximum values we have range $= Max - Min$; the **quartiles**, which divide the data into four equal sized sets - the **median** which divides the data into two equal sized sets is the second quartile, so we have Q_1, Q_2 or median$andQ_3$; and the **IQR** - interquartile range $IQR = Q_3 - Q_1$. We can have a **five number summary** in which we get the $min, Q_1,$ median$, Q_3, max$. Other measures of statistical dispersion include the **variance**, σ^2, and the **standard deviation**, σ. The variance is te mean of the squared differences of the observations to their mean

$$\sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2$$

where N is the number of observations in the sample, $\bar{x}$ is the mean value of the observations and $\{x_1, x_2, \ldots, x_N\}$ are the observed values of the sample items. And the standard deviation is the squared root of the variance

$$\sigma = \sqrt{\sigma^2}$$

[8]:
```python
mydata.describe(include='all')
```

[8]:

	sbp	tobacco	ldl	adiposity	famhist	typea	\
count	462.000000	462.000000	462.000000	462.000000	462	462.000000	
unique	NaN	NaN	NaN	NaN	2	NaN	
top	NaN	NaN	NaN	NaN	Absent	NaN	
freq	NaN	NaN	NaN	NaN	270	NaN	

```
mean     138.326840    3.635649    4.740325   25.406732      NaN   53.103896
std       20.496317    4.593024    2.070909    7.780699      NaN    9.817534
min      101.000000    0.000000    0.980000    6.740000      NaN   13.000000
25%      124.000000    0.052500    3.282500   19.775000      NaN   47.000000
50%      134.000000    2.000000    4.340000   26.115000      NaN   53.000000
75%      148.000000    5.500000    5.790000   31.227500      NaN   60.000000
max      218.000000   31.200000   15.330000   42.490000      NaN   78.000000

           obesity     alcohol         age  chd
count   462.000000  462.000000  462.000000  462
unique         NaN         NaN         NaN    2
top            NaN         NaN         NaN   No
freq           NaN         NaN         NaN  302
mean     26.044113   17.044394   42.816017  NaN
std       4.213680   24.481059   14.608956  NaN
min      14.700000    0.000000   15.000000  NaN
25%      22.985000    0.510000   31.000000  NaN
50%      25.805000    7.510000   45.000000  NaN
75%      28.497500   23.892500   55.000000  NaN
max      46.580000  147.190000   64.000000  NaN
```

By using `describe()`, we can check most of the measures mentioned above. For the numerical variables, we can check the values of the mean and median (2nd Quartile) calculated above. We can check the standard deviation (and therefore the variance) with the formula mentioned, and calculate the range and IQR of each variable. For the nominal variables, we can see what's the most common output and how many times it occurs.

[14]:

```python
#checking the standard deviation
print(mydata.std(),'\n\n')

for i,col in enumerate(numeric_columns):
    sum = 0
    mean = mydata.mean()[i]
    for j in mydata[col]:
        sum += (j - mean) ** 2
    std = np.sqrt(1 / 461 * sum)
    print(col.ljust(10),round(std,6))
```

```
sbp          20.496317
tobacco       4.593024
ldl           2.070909
adiposity     7.780699
typea         9.817534
obesity       4.213680
alcohol      24.481059
age          14.608956
dtype: float64
```

```
sbp          20.496317
tobacco      4.593024
ldl          2.070909
adiposity    7.780699
typea        9.817534
obesity      4.21368
alcohol      24.481059
age          14.608956
```

[26]:
```python
#calculating range and interquartile range
print(''.ljust(10),'range'.ljust(10),'IQR')
for i,col in enumerate(numeric_columns):
    Range = mydata[col].max() - mydata[col].min()
    IQR = mydata[col].quantile(q=0.75) - mydata[col].quantile(q=0.25)
    print(col.ljust(10),str(round(Range,6)).ljust(10),round(IQR,6))
```

```
           range      IQR
sbp        117        24.0
tobacco    31.2       5.4475
ldl        14.35      2.5075
adiposity  35.75      11.4525
typea      65         13.0
obesity    31.88      5.5125
alcohol    147.19     23.3825
age        49         24.0
```

1.3 Outliers

Outliers are datapoint which differ significantly from other observations. They are sometimes excluded from data analysis. There are various ways to define outliers, but commonly, we have that outliers are values displaced from the outers quartiles from 1.5 times the interquartile range, that is

$$x \leq Q_1 - 1.5 \times IQR \vee x \geq Q_3 + 1.5 \times IQR$$

One of the simplest ways to visualise outliers is with the box plot.

1.4 Multivariate Statistics

As the name indicates, multivariate statistics is when we work with more than one variable at a time. Some of the most important measurements of multivariate statistics are the **covariance** which is how much two variables jointly change

$$\mathrm{cov}(X,Y) = \langle (\, X - \langle X \rangle)\,(\,Y - \langle Y \rangle)\,\rangle$$

and the **correlation** is a measure of how much two variable agree with each other

$$\mathrm{corr}(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

where σ_X and σ_Y are the standard deviations of the two variables X and Y (Wikipedia). The correlation matrix is symmetric and its diagonal values are all 1, that is, all variables have maximum correlation with themselves.

[28]:
```
#we can check the properties above by getting the correlation matrix of the␣
↪variables in this dataset
mydata.corr()
```

[28]:
```
                sbp   tobacco       ldl  adiposity     typea   obesity  \
sbp        1.000000  0.212247  0.158296   0.356500 -0.057454  0.238067
tobacco    0.212247  1.000000  0.158905   0.286640 -0.014608  0.124529
ldl        0.158296  0.158905  1.000000   0.440432  0.044048  0.330506
adiposity  0.356500  0.286640  0.440432   1.000000 -0.043144  0.716556
typea     -0.057454 -0.014608  0.044048  -0.043144  1.000000  0.074006
obesity    0.238067  0.124529  0.330506   0.716556  0.074006  1.000000
alcohol    0.140096  0.200813 -0.033403   0.100330  0.039498  0.051620
age        0.388771  0.450330  0.311799   0.625954 -0.102606  0.291777

            alcohol       age
sbp        0.140096  0.388771
tobacco    0.200813  0.450330
ldl       -0.033403  0.311799
adiposity  0.100330  0.625954
typea      0.039498 -0.102606
obesity    0.051620  0.291777
alcohol    1.000000  0.101125
age        0.101125  1.000000
```

1.5 Graphical Displays

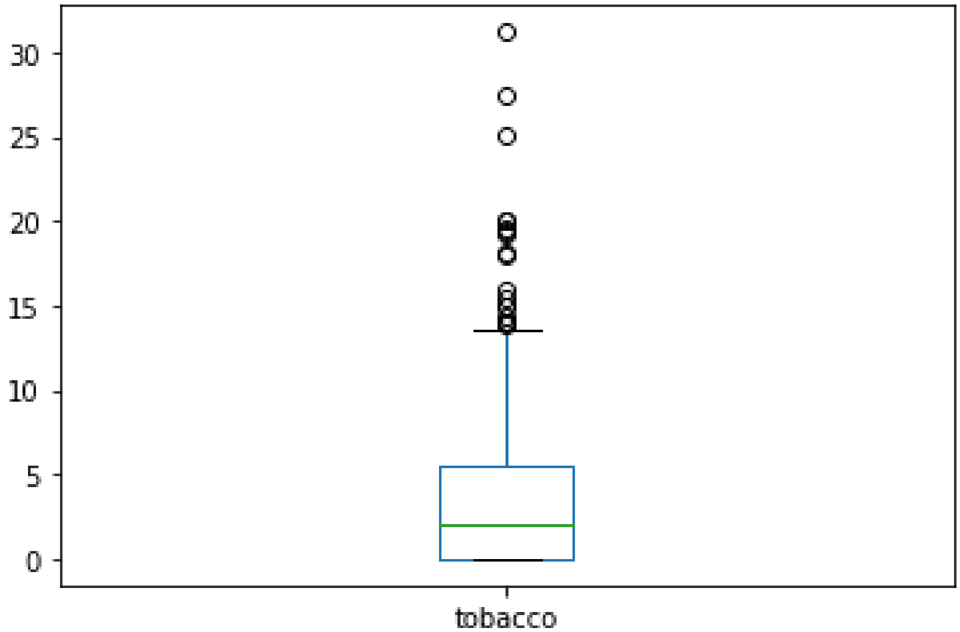
Pandas has several ways to display graphics of our data that are useful in statistical analysis. Here we explore the box plot, the bar plot, the pie plot, the histogram, the density plot, the quantile plot, and for multivariate statistics, the scatter plot.

1.5.1 Box plot

Displays the dataset based on the five-number summary described above, that is, the minimum, maximum, and the 3 quartiles. The minimum and maximum displayed do not include outliers, so it's a clear way to see them.

[32]:
```
#lets check for our data, say for the tobbaco column
mydata['tobacco'].plot.box()
#we can see several outliers above the maximum value
```

[32]: <AxesSubplot:>

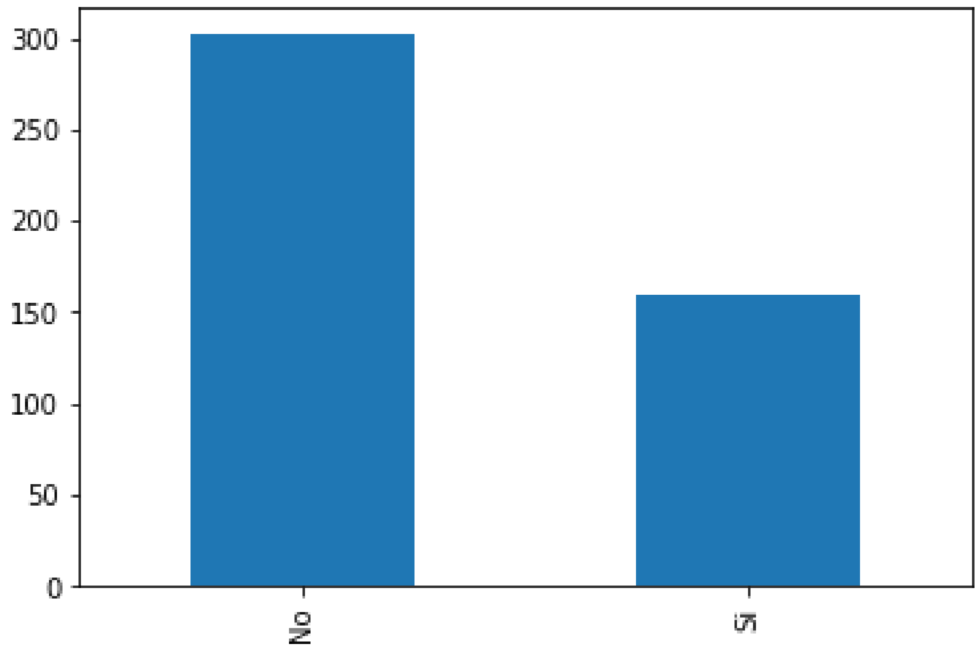


1.5.2 Bar plot

The bar plot presents categorical data with rectangular bars with heights proportional to the values they represent.

[34]:
```
#example for 'chd'
mydata['chd'].value_counts().plot.bar()
```

[34]: <AxesSubplot:>

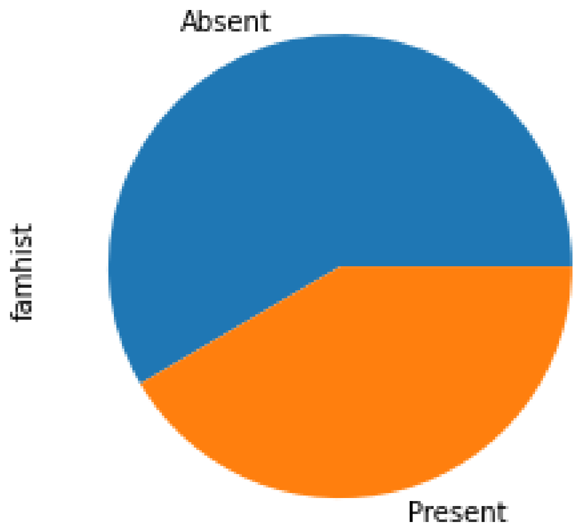


1.5.3 Pie plot

A pie plot is a circular display of data to illustrate numerical proportions. The arc length of each slice is proportional to the quantity it represents.

[36]:
```
#lets check for 'famhist' this time
mydata['famhist'].value_counts().plot.pie()
```

[36]: <AxesSubplot:ylabel='famhist'>

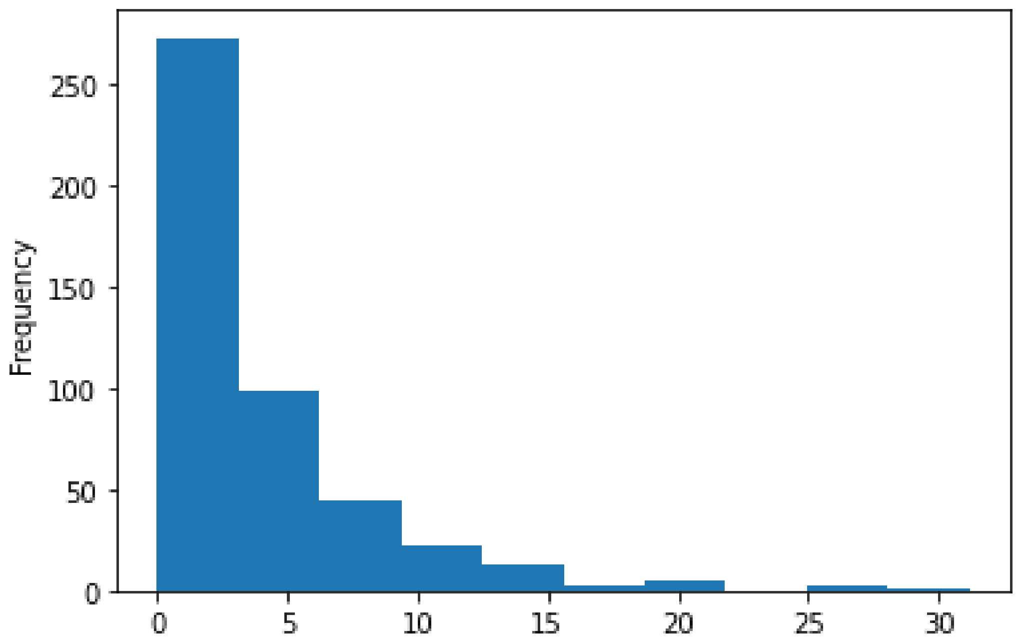


1.5.4 Histogram

A histogram is an approximate representation of th distribution of numerical data, where we divide the ranf of values into a series of intervals - bins - and count how many values fall into each interval.

[37]:
```
#lets see an histogram for the 'tobacco' variable
mydata['tobacco'].plot.hist()
```

[37]: <AxesSubplot:ylabel='Frequency'>

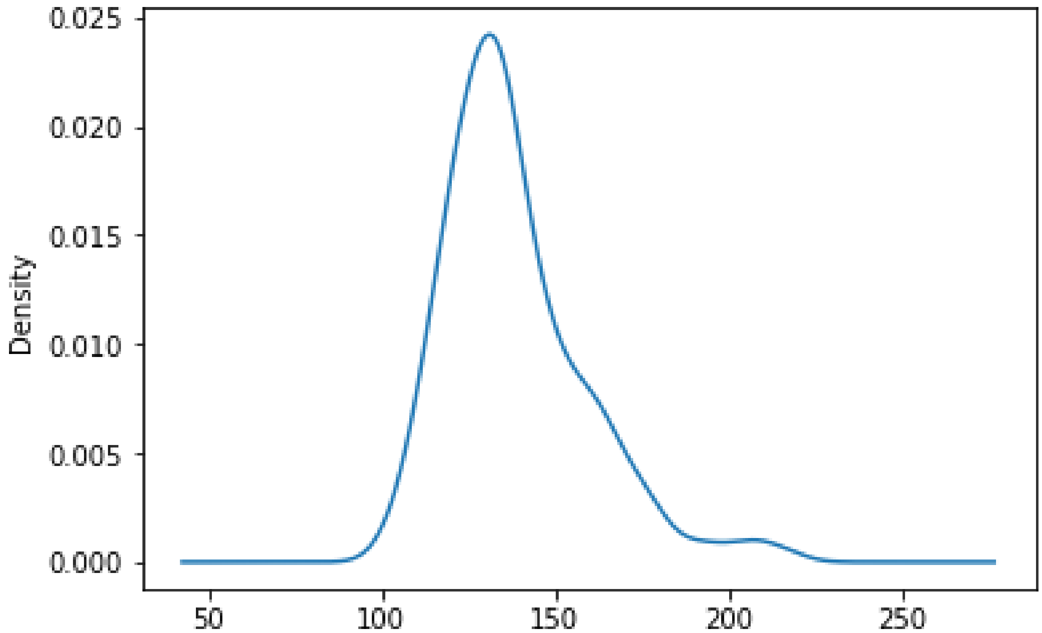


1.5.5 Density plot

A density plot is a plot used to observe the distribution of a variable in a dataset. It tries to estimate the probability density funcion of the variable, using kernel density estimates.

[39]:
```
#example for the 'sbp' variable
mydata['sbp'].plot.density()
```

[39]:
```
<AxesSubplot:ylabel='Density'>
```

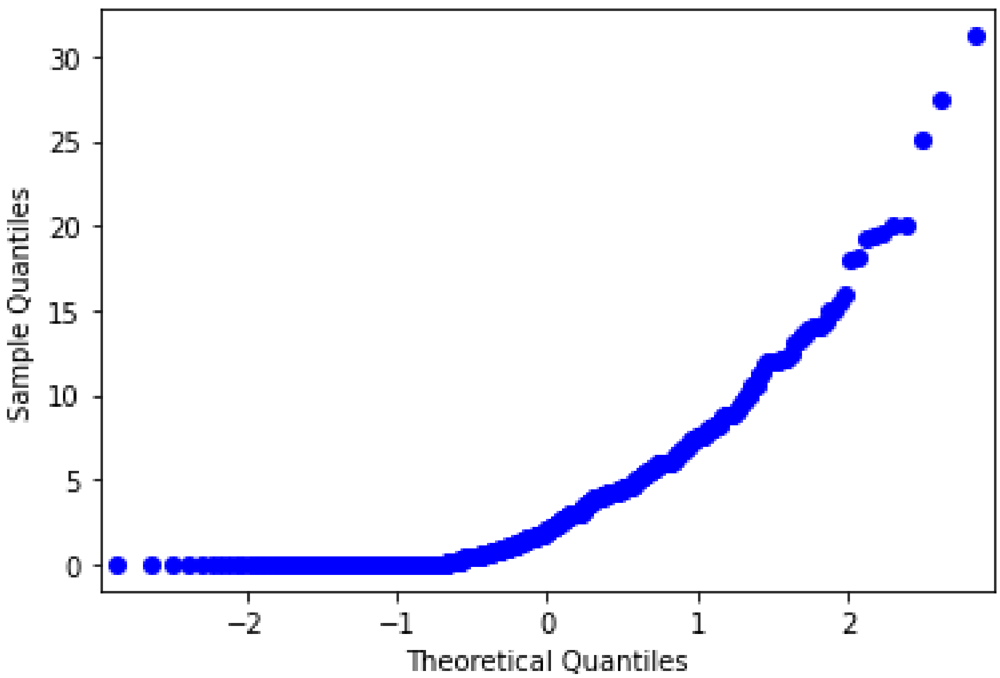


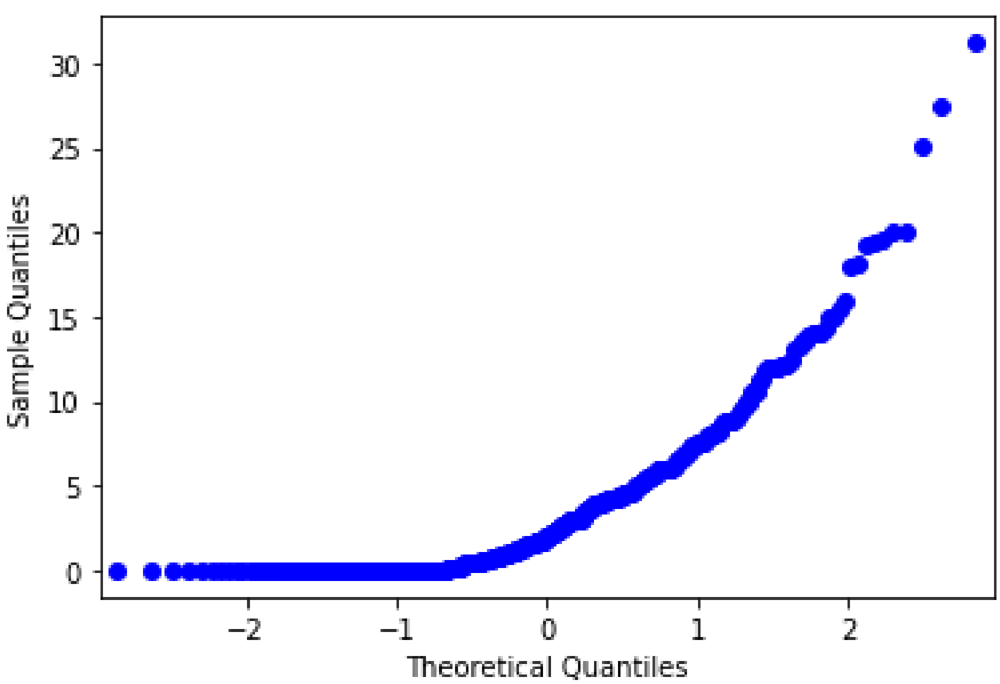
1.5.6 Quantile plot

The quantile plot is a graphical method of comparing two probability distributions, by plotting their quantiles against each other.

[45]:
```
#example with the 'tobacco' variable
sm.qqplot(mydata['tobacco'])
```

[45]:


Sample Quantiles
Theoretical Quantiles
0
5
10
15
20
25
30
−2
−1
0
1
2

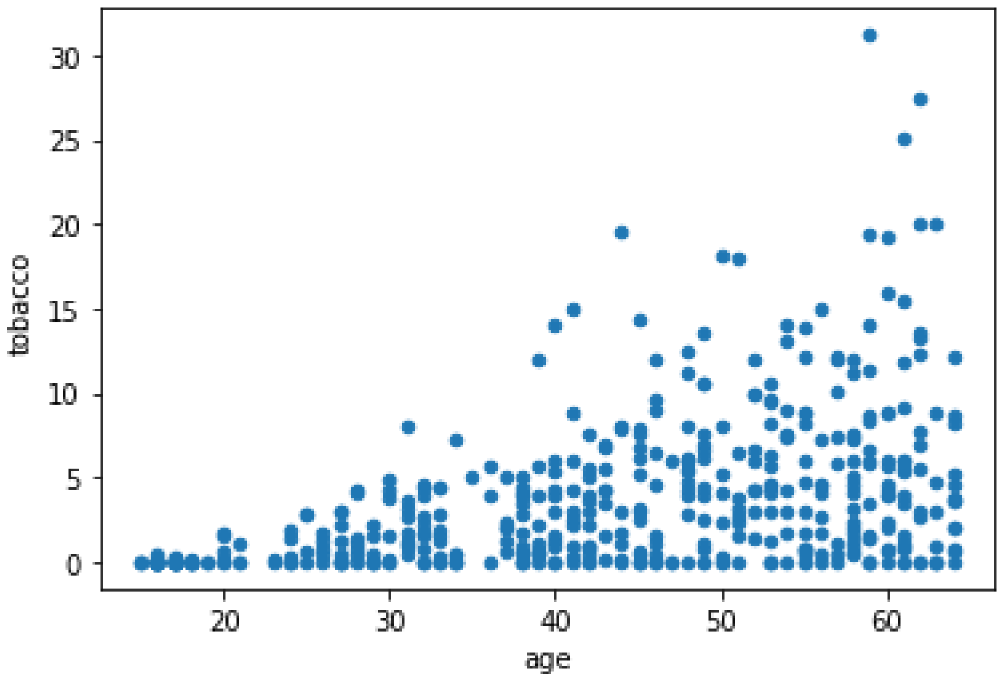

Sample Quantiles
Theoretical Quantiles
0
5
10
15
20
25
30
−2
−1
0
1
2

1.5.7 Scatter plots

Scatter plots use Cartesian coordinates to display values for typically two variables for a set of data. Particularly useful for multivariate statistics.

[46]:
```
#example for 'age' and 'tobacco'
mydata.plot.scatter('age','tobacco')
```

[46]: <AxesSubplot:xlabel='age', ylabel='tobacco'>



1.6 References

As references in this work, I used Wikipedia and material from the class lectures.