1.

[5, 20, 1, 6, 13, 8, 9, 11, 17, 7, 2, 12]

1. **Equal-frequency binning**

Sorted:

[1, 2, 5, 6, 7, 8, 9, 11, 12, 13, 17, 20]

Bins

Bin1: 1, 2, 5, 6

Bin2: 7, 8, 9, 11

Bin3: 12, 13, 17, 20

2. **Smoothing by bin boundaries**

Before Bin Boundaries:

Bin1: 1, 2, 5, 6

Bin2: 7, 8, 9, 11

Bin3: 12, 13, 17, 20

After bin boundaries:

Bin1: 1, 1, 6, 6

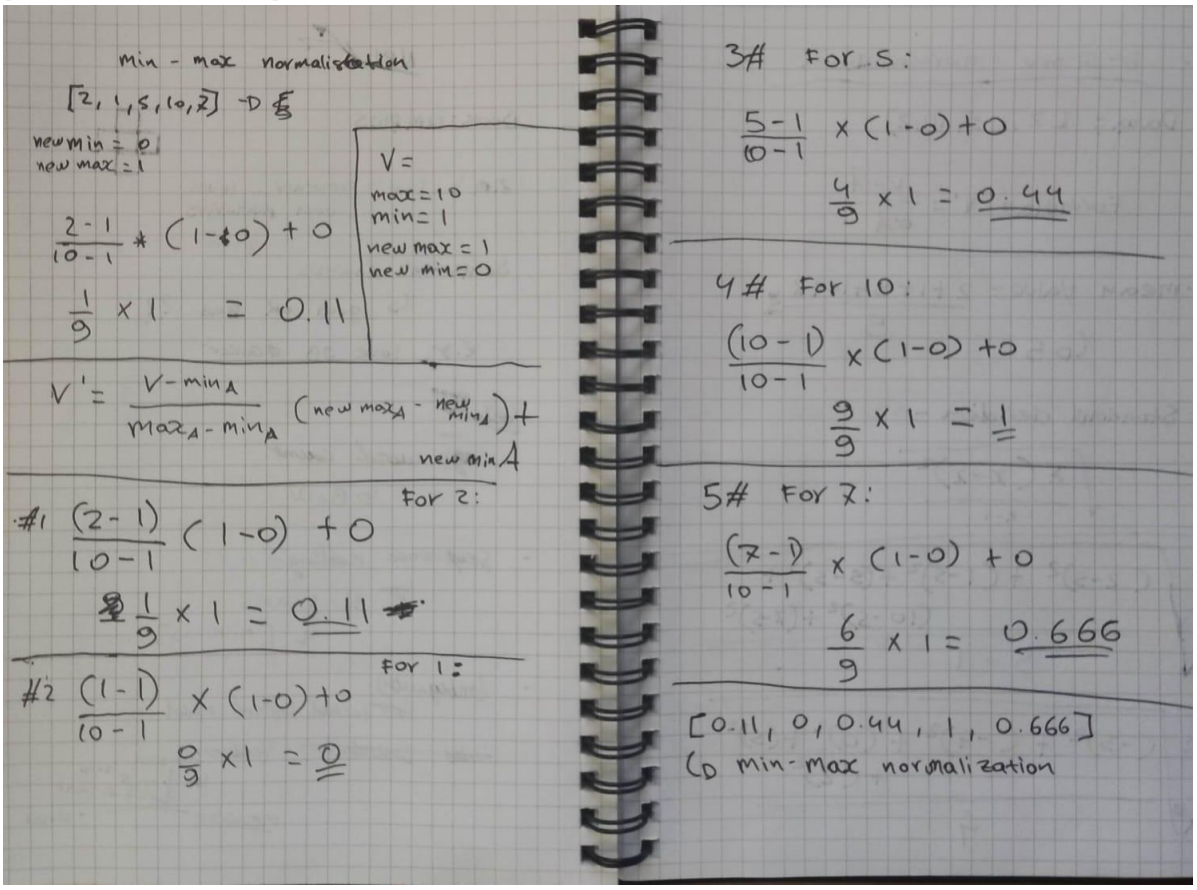
Bin2: 7, 7, 7, 11

Bin3: 12, 12, 20, 20

2- Use the below methods to normalize the following data: [2, 1, 5, 10, 7]:

1. **min-max normalization with min=0 and max=1.**

[0.11, 0, 0.44, 1, 0.666]



2. **z-score normalization**

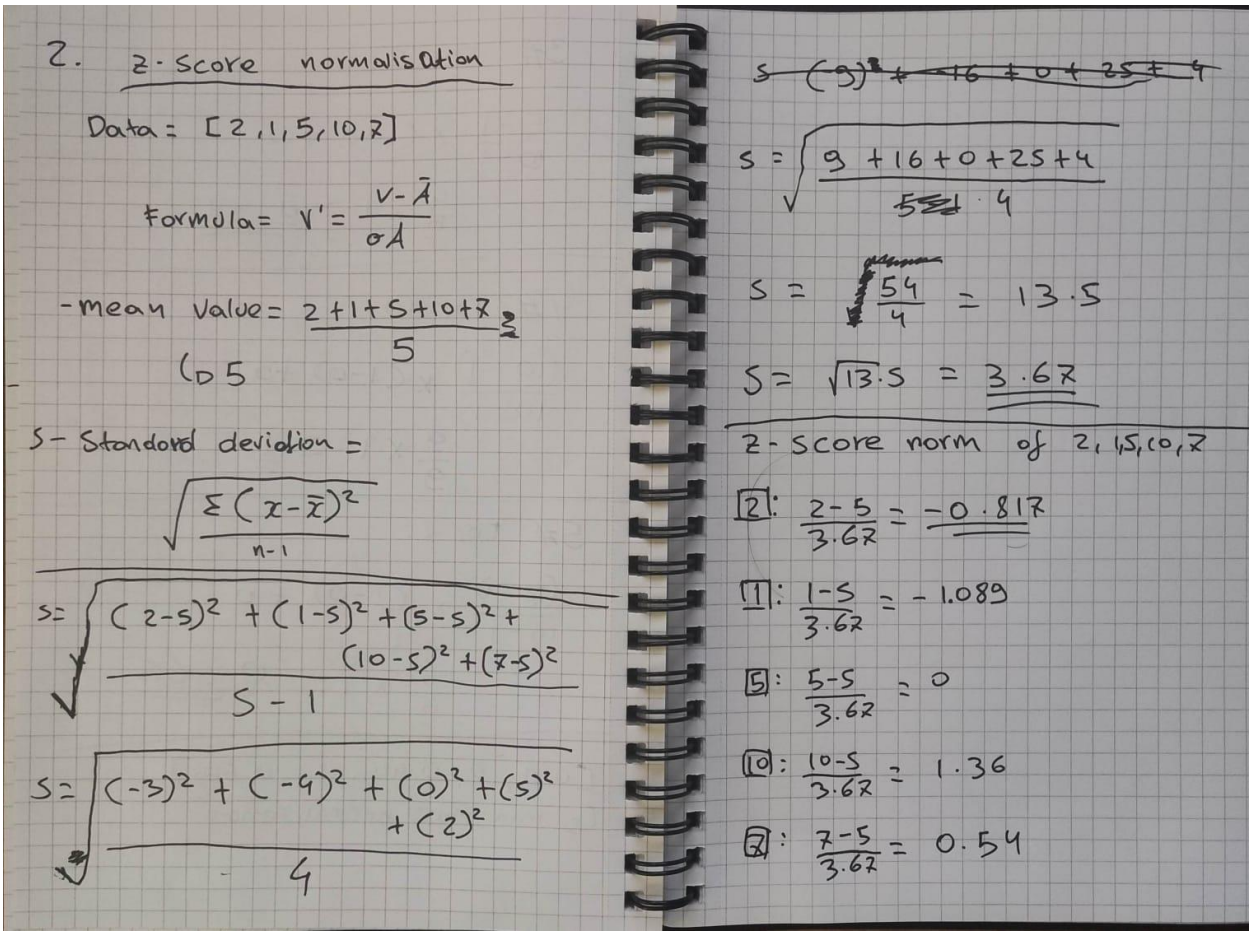
   2 = -0.817

   1 = -1.089

   5 = 0

   10 = 1.36

   7 = 0.54



3. **Students at two universities, University A and University B, have been provided with feedback forms on student satisfaction, with the below responses recorded. Is student satisfaction correlated with a specific university? Use a chi-square test to find out, assuming a significance level of 0.001 and a corresponding chi-square significance value of 10.828. [1 mark out of 5]**

Observation

	University A	University B	
satisfied	47	86	133
dissatisfied	25	49	74
total:	72	135	207

Expected Values

	University A	University B		
satisfied	46.2608696	86.73913		
dissatisfied	25.7391304	48.26087		
total:				
(O-E)^2	0.5463138	0.5463138	0.54631	0.54631
((O-E)^2)/E	0.01180941	0.0062984	0.02123	0.01132
SUM:	0.05065281			

((O-E)^2)/E	University A	University B	
satisfied	0.01180941	0.0062984	
dissatisfied	0.02122503	0.01132	
total:			
x^2	0.05065281		
df =	1		
p-value	0.8219312		
chi test	0.8219312		

No there isn't correlation between student satisfaction with a specific university.

4. Data Cleaning

part 4

sk-learn approach

[6]
```
import pandas as pd
import numpy as np
data = pd.read_csv('country-income.csv')
```

[7]
```
data.head()
```

	Region	Age	Income	Online Shopper
0	India	49.0	86400.0	No
1	Brazil	32.0	57600.0	Yes
2	USA	35.0	64800.0	No
3	Brazil	43.0	73200.0	No
4	USA	45.0	NaN	Yes

[8]
```
data = data.iloc[:, :].values
```

[9]
```
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
imputer.fit(data[:, 1:3])
data[:, 1:3] = imputer.transform(data[:, 1:3])
```

[10]
```
data = pd.DataFrame(data, columns=["Region", "Age", "Income", "Online Shopper"])
```

[11]
```
data.head()
```

	Region	Age	Income	Online Shopper
0	India	49.0	86400.0	No
1	Brazil	32.0	57600.0	Yes
2	USA	35.0	64800.0	No
3	Brazil	43.0	73200.0	No
4	USA	45.0	76533.333333	Yes

pandas traditional approach

```
[12] data = pd.read_csv('country-income.csv')
```

```
[13] data["Age"] = data["Age"].fillna(data["Age"].mean())
     data["Income"] = data["Income"].fillna(data["Income"].mean())
```

```
[14] data.head()
```

	Region	Age	Income	Online Shopper
0	India	49.0	86400.000000	No
1	Brazil	32.0	57600.000000	Yes
2	USA	35.0	64800.000000	No
3	Brazil	43.0	73200.000000	No
4	USA	45.0	76533.333333	Yes

5. Plot scatterplot of shoe size…

Question 5

```
[15] import pandas as pd
     data = pd.read_csv('shoesize.csv')
```

```
[16] data.head()
```

	Index	Gender	Size	Height
0	1	F	5.5	60.0
1	2	F	6.0	60.0
2	3	F	7.0	60.0
3	4	F	8.0	60.0
4	5	F	8.0	60.0

For Male:

▾ for Male

```
[17] gender = {'M': 1,'F': 0}
```

```
[18] data.Gender = [gender[item] for item in data.Gender]
```

```
[19] data.head()
```

	Index	Gender	Size	Height
0	1	0	5.5	60.0
1	2	0	6.0	60.0
2	3	0	7.0	60.0
3	4	0	8.0	60.0
4	5	0	8.0	60.0

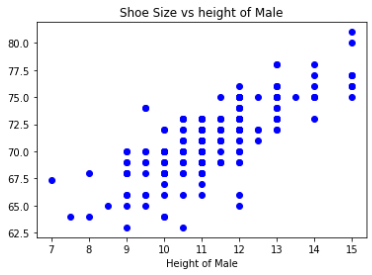
```
[20] rslt_df = data[(data['Gender']==1)]
```

```
[21] rslt_df.head()
```

	Index	Gender	Size	Height
187	188	1	10.5	63.0
188	189	1	9.0	63.0
189	190	1	7.5	64.0
190	191	1	8.0	64.0
191	192	1	10.0	64.0

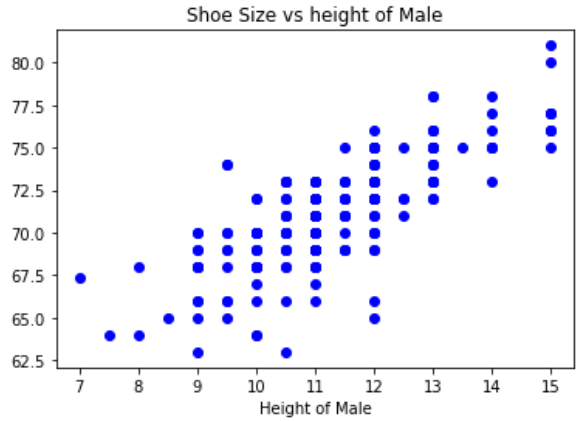
```
[22] import matplotlib.pyplot as plt
     plt.scatter(rslt_df["Size"], rslt_df["Height"], c ="blue")
     plt.title("Shoe Size vs height of Male")
     plt.xlabel("Shoe Size")
     plt.xlabel("Height of Male")

     # To show the plot
     plt.show()
```



```
import matplotlib.pyplot as plt
plt.scatter(rslt_df["Size"], rslt_df["Height"], c ="blue")
plt.title("Shoe Size vs height of Male")
plt.xlabel("Shoe Size")
plt.xlabel("Height of Male")

# To show the plot
plt.show()
```




```
[23] # Getting the Pearson Correlation Coefficient
     correlation = rslt_df.corr()
     print(correlation.loc['Size', 'Height'])
```

```
0.7677093547300965
```

For Female:

for female

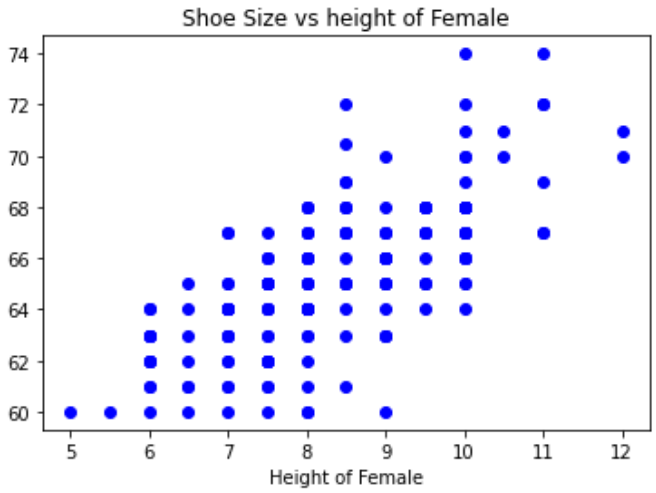
```
[24] rslt_df = data[(data['Gender']==0)]
```

```
[25] rslt_df.head()
```

	Index	Gender	Size	Height
0	1	0	5.5	60.0
1	2	0	6.0	60.0
2	3	0	7.0	60.0
3	4	0	8.0	60.0
4	5	0	8.0	60.0

```
[26] import matplotlib.pyplot as plt
     plt.scatter(rslt_df["Size"], rslt_df["Height"], c ="blue")
     plt.title("Shoe Size vs height of Female")
     plt.xlabel("Shoe Size")
     plt.xlabel("Height of Female")

     # To show the plot
     plt.show()
```




```
[27] # Getting the Pearson Correlation Coefficient
     correlation = rslt_df.corr()
     print(correlation.loc['Size', 'Height'])
```

```
0.7078119417143964
```

Scatter plot and Pearson Correlation Coefficient tells us that height of male has more direct relation with shoe size than the height of female.

Q6.

```
[ ] import pandas as pd
    data = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data', header=None)
    data.columns = ['Sample code', 'Clump Thickness', 'Uniformity of Cell Size', 'Uniformity of Cell Shape',
                    'Marginal Adhesion', 'Single Epithelial Cell Size', 'Bare Nuclei', 'Bland Chromatin',
                    'Normal Nucleoli', 'Mitoses','Class']

    data = data.drop(['Sample code'],axis=1)
    data.head()
```

	Clump Thickness	Uniformity of Cell Size	Uniformity of Cell Shape	Marginal Adhesion	Single Epithelial Cell Size	Bare Nuclei	Bland Chromatin	Normal Nucleoli	Mitoses	Class
0	5	1	1	1	2	1	3	1	1	2
1	5	4	4	5	7	10	3	2	1	2
2	3	1	1	1	2	2	3	1	1	2
3	6	8	8	1	3	4	3	7	1	2
4	4	1	1	3	2	1	3	1	1	2

```
data["Class"].unique()
```

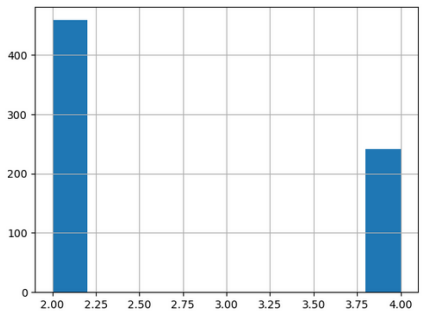
```
array([2, 4], dtype=int64)
```

```
[ ] data["Class"].value_counts()
```

```
2    458
4    241
Name: Class, dtype: int64
```

```
[ ] data["Class"].hist()
```

```
<AxesSubplot:>
```



So we can clearly see that Dataset is not balanced as value counts for our target variable has a significant different so we should startified=True when we will do train test spliting for our dataset and same goes for PCA so that we can conserve and maintain class balance in our data.

Assignment 1 [part 2 of 2]

1. In Section 1, what kind of relationship can be inferred from summary statistics regarding `ACT composite score` and `SAT total score`? Which visualisations make this relationship apparent? [0.5 marks out of 5]

As the ACT composite score increases, the SAT total score also increases. Students who have a median SAT score of 2000 will have more than 25+ in ACT composite score. Scatter plots can make the visualisation apparent.

2. Based on the box plots presented in Section 1, what is the relationship between `parental level of education` and `parental income`? Using table visualisation, find and show the entire rows that correspond to the outliers regarding `parental income` whose parents have a master's degree. [0.5 marks out of 5]

The median of the parental level of education increased as along side the parental income. Each level of education had higher income that it's lower-level counterpart.

	ACT composite score	SAT total score	parental level of education	parental income	high school gpa	college gpa	years to graduate
26	31	2108	master's degree	120391	4.0	3.6	4
29	28	2097	master's degree	59724	3.9	3.2	4

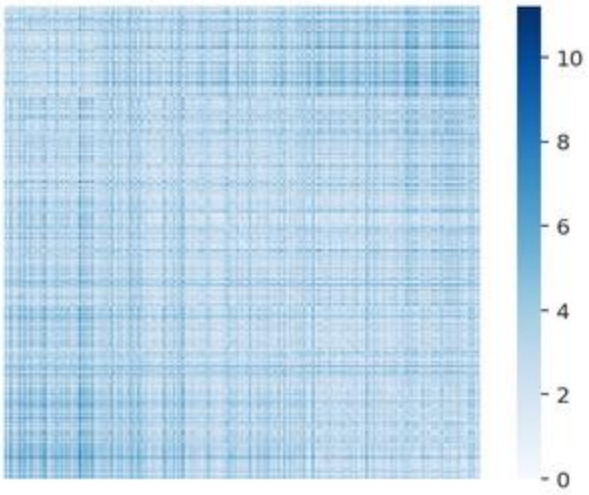
3. Using an example, explain the importance of scaling features so that their magnitudes are comparable when computing distances. [0.5 marks out of 5]

Scaling features are essential machine learning algorithms for calculating distances between data. As an example, if there is a vast difference in the range of a dataset between 10s and 1000s and it makes assumption of higher values to be superior. It can affect the training of a model as it would start to put these higher values to have significant roles in the model.

As example, someone had a college GPA of 3.7 and the years to graduate was 3 years. Both numbers are similar in range but represent different things but the model as a feature it treats it same. Considering the GPA and the graduation time as features the algorithm would make it, so the GPA has a higher value than the graduation time thus the GPA being more important. However, if we make the graduation time (years) into months it becomes 36 months which is more dominant than the GPA.

For this reason, feature scaling is required to bring every feature in the similar context or equality to have better scalability for the range of data.

4. In Section 1, the distance matrix visualization is not very informative. However, it is still possible to infer that the average distance between students whose parents only have some high school education and students whose parents have a master's degree is larger than the average distance between students whose parents only have some high school education. Explain how this inference is possible from the visualization. [0.5 marks out of 5]



From the distance matrix visualization, the distance for some high school education is far left bottom which contrasts with whose parents have a master's degree which is much higher coming far right top of the matrix. The density in the lower left part is much greater suggesting the average for some high school education. As for the masters the density can be seem much greater at the highest point of 10 at average.

5. In Section 2, increase the number of evenly spaced numbers from 10 to 100 for both axes and observe the corresponding heat map created through nearest neighbor interpolation. Read about this interpolation method and explain what you observed. [0.5 marks out of 5]

```python
import numpy as np
x_range = np.linspace(10, 100, 10)
y_range = np.linspace(10, 100, 10)

# meshgrid: X[i, j] == x_range[j] and Y[i, j] == y_range[i]
X, Y = np.meshgrid(x_range, y_range)

# Z[i, j] == f(x_range[j], y_range[i])
Z = X**2 + Y**2

# Dataset representation
df = pd.DataFrame({'x': X.reshape(-1), 'y': Y.reshape(-1), 'z = f(x,y)': Z.reshape(-1)})
display(df)
```

	x	y	z = f(x,y)
0	10.0	10.0	200.0
1	20.0	10.0	500.0
2	30.0	10.0	1000.0
3	40.0	10.0	1700.0
4	50.0	10.0	2600.0
...	...	...	...
95	60.0	100.0	13600.0
96	70.0	100.0	14900.0
97	80.0	100.0	16400.0
98	90.0	100.0	18100.0
99	100.0	100.0	20000.0

100 rows × 3 columns

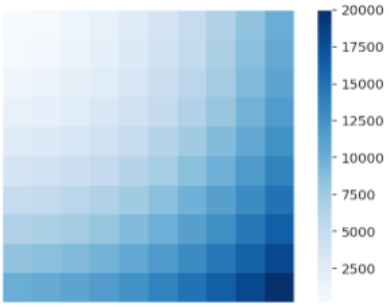
2.1 Heat maps

The `matplotlib` function `imshow` can be used to create a heatmap through nearest neighbour interpolation.

```
[ ]  # Interpolation: point (x, y) is colored according to the value z of the nearest point in the dataset
     plt.imshow(Z, cmap='Blues', aspect='equal', interpolation='nearest')
     plt.colorbar()

     # xticks and yticks would show Z matrix indices
     plt.xticks([])
     plt.yticks([])

     plt.show()
```



Nearest Neighbor interpolation is one of simplest type of interpolation which requires little calculations allowing it to be one of fastest algorithms. The Nearest Neighbor interpolation determines its pixel based on the closest data value in that position.

From the observation, the distribution of pixel is about equal for both axes. This is because it's evenly spaced.

6. Wine Data

6.1 Load the wine dataset. Compute the frequency of each value of the 'target' feature. [0.5 marks out of 5]

```
from sklearn.datasets import load_wine

data = load_wine()

df = pd.DataFrame(data.data, columns = data.feature_names)

df['target'] = pd.Series(data.target)

df
```

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity	hue	od280/od315_of_diluted_wines	proline	target
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065.0	0
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050.0	0
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185.0	0
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480.0	0
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735.0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
173	13.71	5.65	2.45	20.5	95.0	1.68	0.61	0.52	1.06	7.70	0.64	1.74	740.0	2
174	13.40	3.91	2.48	23.0	102.0	1.80	0.75	0.43	1.41	7.30	0.70	1.56	750.0	2
175	13.27	4.28	2.26	20.0	120.0	1.59	0.69	0.43	1.35	10.20	0.59	1.56	835.0	2
176	13.17	2.59	2.37	20.0	120.0	1.65	0.68	0.53	1.46	9.30	0.60	1.62	840.0	2
177	14.13	4.10	2.74	24.5	96.0	2.05	0.76	0.56	1.35	9.20	0.61	1.60	560.0	2

178 rows × 14 columns

```
[ ]  df.target = df.target.astype('int64').astype('category')

     #freq

     frquency = df['target'].value_counts()

     frquency

     # frquency.plot(kind = 'bar')
```

```
1    71
0    59
2    48
Name: target, dtype: int64
```