Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Технологии машинного обучения»

Лабораторная работа №1-2

Выполнил:
студент группы ИУ5-64Б
Коваленко Г. В.

Проверил:
Гапанюк Ю. Е.

2024 г.

Качество яблок

Описание:

Этот набор данных содержит информацию о различных атрибутах набора фруктов, дающую представление об их характеристиках. Набор данных включает такие сведения, как идентификатор плода, размер, вес, сладость, хрусткость, сочность, зрелость, кислотность и качество.

[Ссылка на датасет]

```
!unzip archive.zip
```

```
Archive:  archive.zip
  inflating: 8000_popular_tracks.csv
```

```
!unzip "archive(1).zip"
```

```
Archive:  archive(1).zip
  inflating: apple_quality.csv
```

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sb
from sklearn.preprocessing import LabelEncoder
```

```
df_apple = pd.read_csv("apple_quality.csv",
                       sep=",",
                       low_memory=False)
```

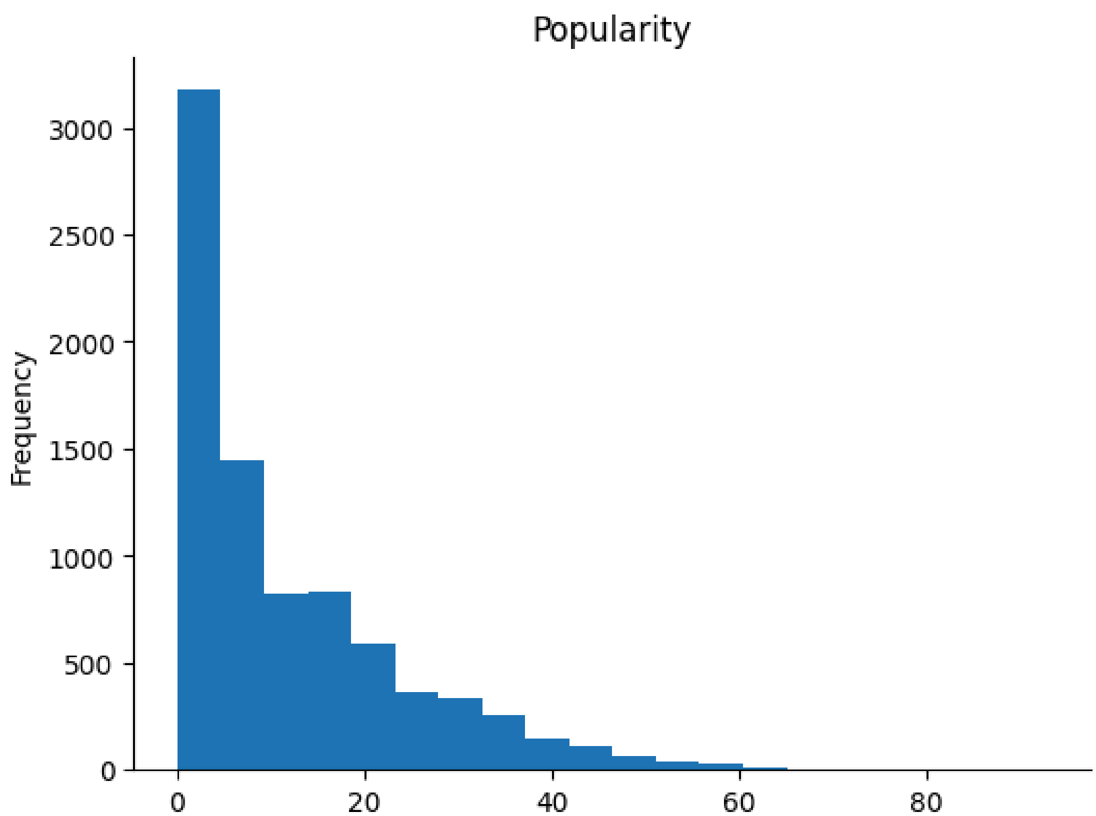
```
df_apple.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4001 entries, 0 to 4000
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   A_id         4000 non-null   float64
 1   Size         4000 non-null   float64
 2   Weight       4000 non-null   float64
 3   Sweetness    4000 non-null   float64
 4   Crunchiness  4000 non-null   float64
 5   Juiciness    4000 non-null   float64
 6   Ripeness     4000 non-null   float64
 7   Acidity      4001 non-null   object
 8   Quality      4000 non-null   object
dtypes: float64(7), object(2)
memory usage: 281.4+ KB
```

```
df_apple.head()
```

	A_id	Size	Weight	Sweetness	Crunchiness	Juiciness	Ripeness	Acidity	Quality
0	0.0	-3.970049	-2.512336	5.346330	-1.012009	1.844900	0.329840	-0.491590483	good
1	1.0	-1.195217	-2.839257	3.664059	1.588232	0.853286	0.867530	-0.722809367	good
2	2.0	-0.292024	-1.351282	-1.738429	-0.342616	2.838636	-0.038033	2.621636473	bad
3	3.0	-0.657196	-2.271627	1.324874	-0.097875	3.637970	-3.413761	0.790723217	good
4	4.0	1.364217	-1.296612	-0.384658	-0.553006	3.030874	-1.303849	0.501984036	good

```
# @title Popularity

from matplotlib import pyplot as plt
df_tracks['Popularity'].plot(kind='hist', bins=20, title='Popularity')
plt.gca().spines[['top', 'right',]].set_visible(False)
```




```
df_apple.describe()
```

	A_id	Size	Weight	Sweetness	Crunchiness	Juiciness	Ripeness
count	4000.000000	4000.000000	4000.000000	4000.000000	4000.000000	4000.000000	4000.000000
mean	1999.500000	-0.503015	-0.989547	-0.470479	0.985478	0.512118	0.498277
std	1154.844867	1.928059	1.602507	1.943441	1.402757	1.930286	1.874427
min	0.000000	-7.151703	-7.149848	-6.894485	-6.055058	-5.961897	-5.864599
25%	999.750000	-1.816765	-2.011770	-1.738425	0.062764	-0.801286	-0.771677
50%	1999.500000	-0.513703	-0.984736	-0.504758	0.998249	0.534219	0.503445
75%	2999.250000	0.805526	0.030976	0.801922	1.894234	1.835976	1.766212
max	3999.000000	6.406367	5.790714	6.374916	7.619852	7.364403	7.237837

```
df_apple.isnull().sum()
```

```
A_id           1
Size           1
Weight         1
Sweetness      1
Crunchiness    1
Juiciness      1
Ripeness       1
Acidity        0
Quality        1
dtype: int64
```

```
df_apple.duplicated().sum()
```

```
0
```

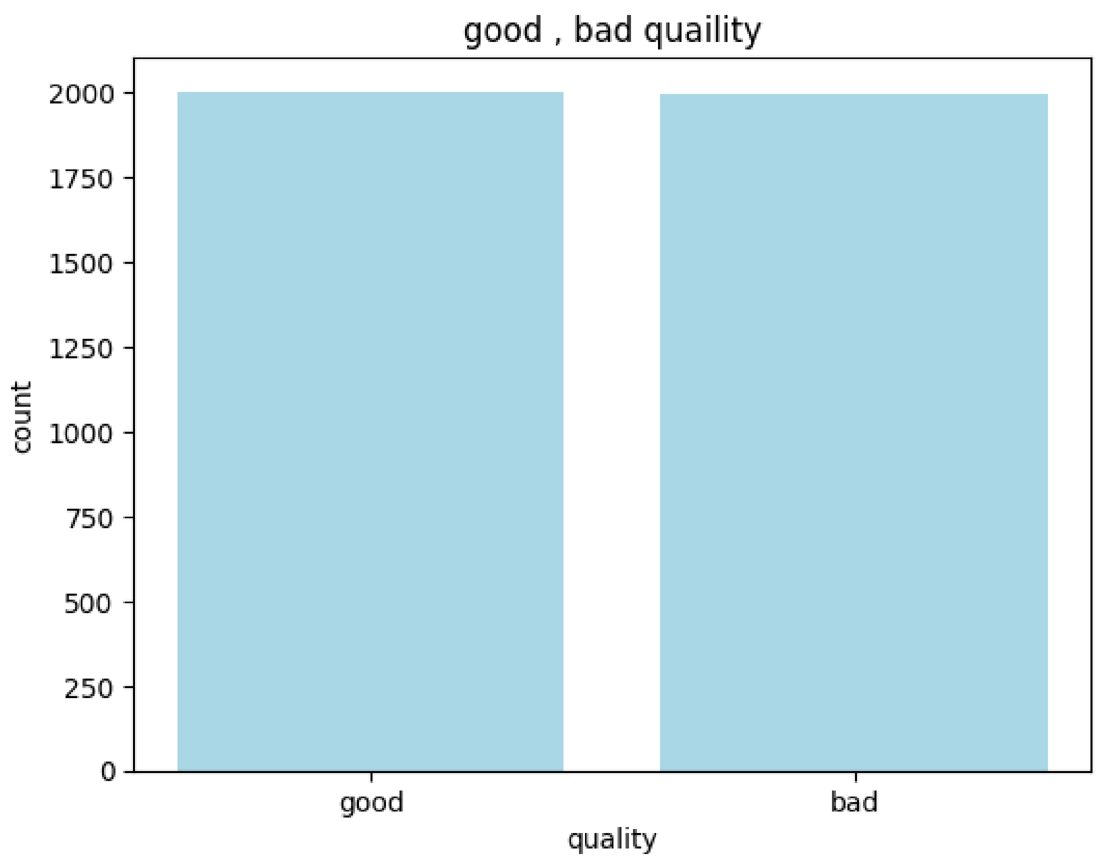
```
df_apple=df_apple.dropna()
df_apple.head()
```

	A_id	Size	Weight	Sweetness	Crunchiness	Juiciness	Ripeness	Acidity	Quality
0	0.0	-3.970049	-2.512336	5.346330	-1.012009	1.844900	0.329840	-0.491590483	good
1	1.0	-1.195217	-2.839257	3.664059	1.588232	0.853286	0.867530	-0.722809367	good
2	2.0	-0.292024	-1.351282	-1.738429	-0.342616	2.838636	-0.038033	2.621636473	bad
3	3.0	-0.657196	-2.271627	1.324874	-0.097875	3.637970	-3.413761	0.790723217	good
4	4.0	1.364217	-1.296612	-0.384658	-0.553006	3.030874	-1.303849	0.501984036	good

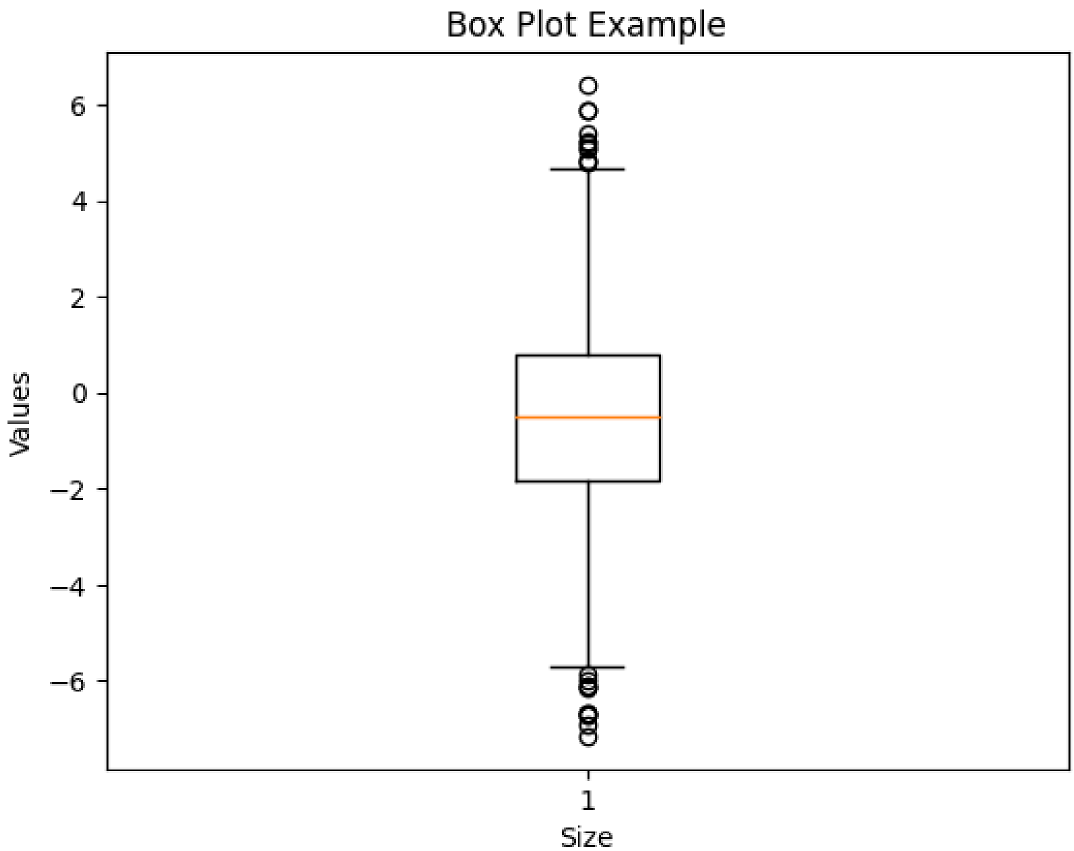
```
compare = df_apple['Quality'].value_counts()
compare
```

```
good    2004
bad     1996
Name: Quality, dtype: int64
```

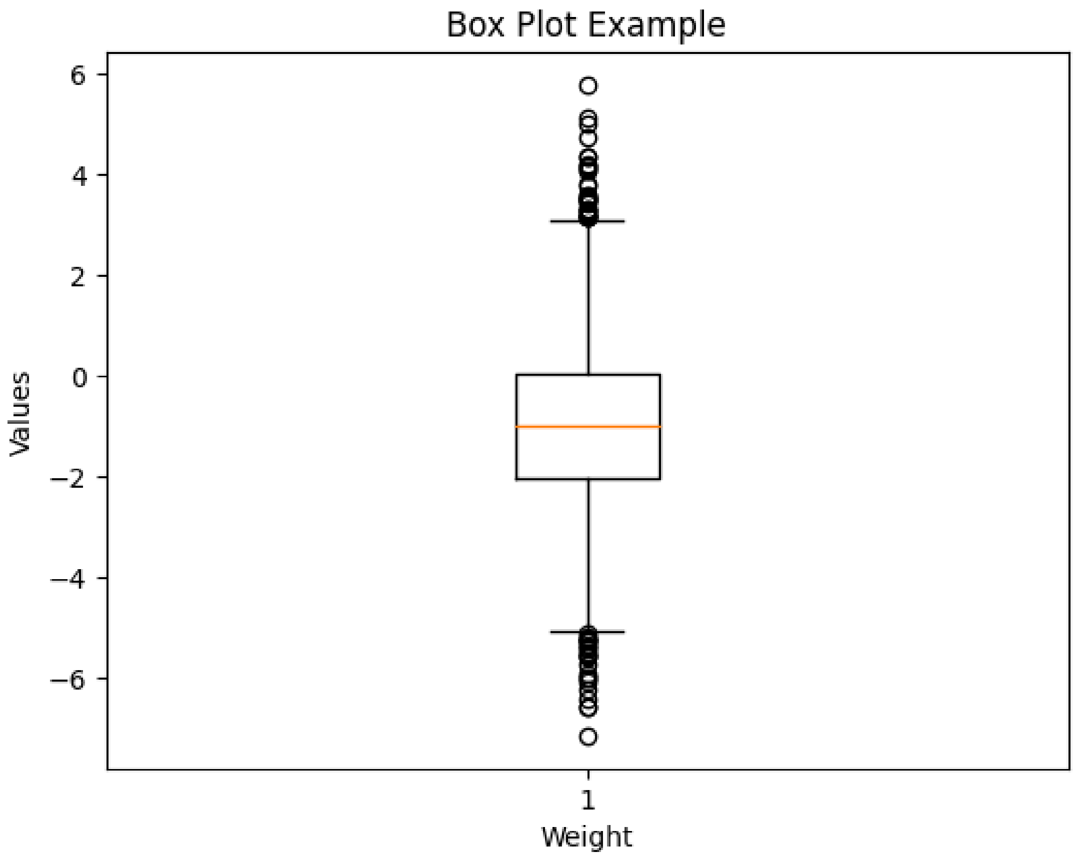
```
plt.bar(compare.index, compare, color='lightblue')

plt.xlabel('quality')
plt.ylabel('count')
plt.title('good , bad quaility')

plt.show()
```



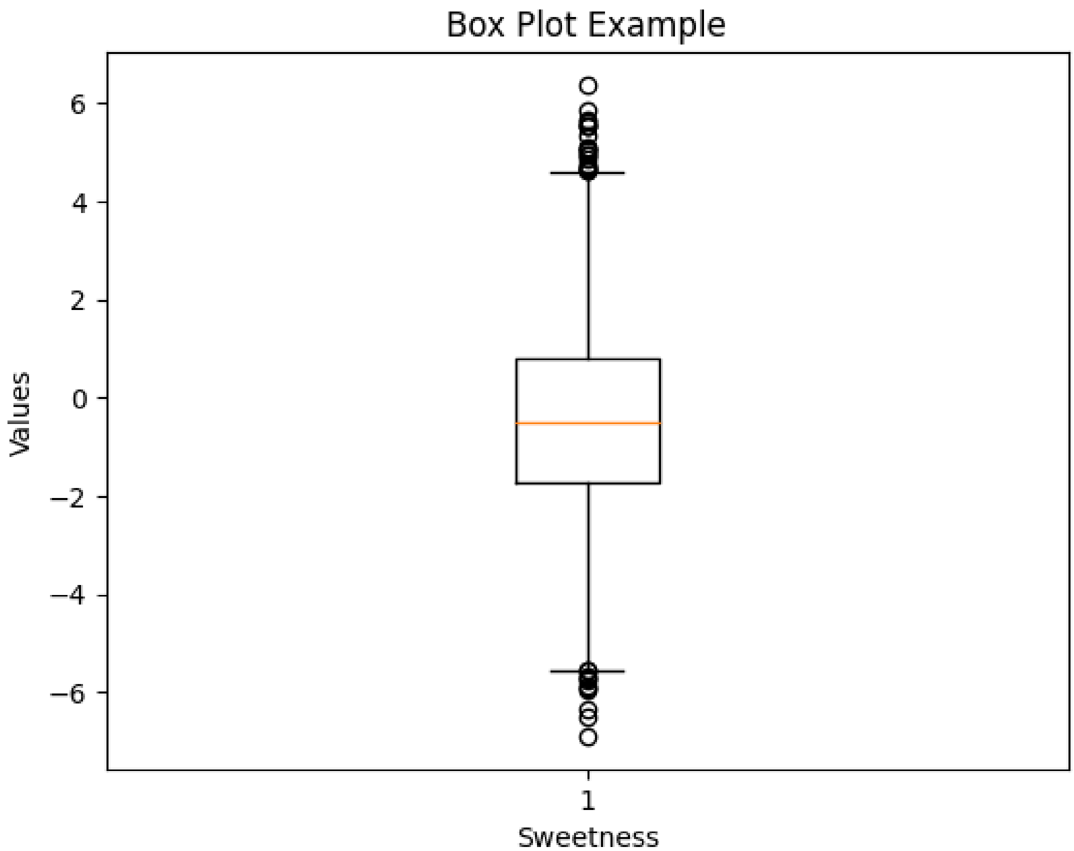

```
plt.boxplot(df_apple['Size'])

plt.xlabel('Size')
plt.ylabel('Values')
plt.title('Box Plot Example')

plt.show()
```




```
plt.boxplot(df_apple['Weight'])

plt.xlabel('Weight')
plt.ylabel('Values')
plt.title('Box Plot Example')

plt.show()
```




```
plt.boxplot(df_apple['Sweetness'])

plt.xlabel('Sweetness')
plt.ylabel('Values')
plt.title('Box Plot Example')

plt.show()
```

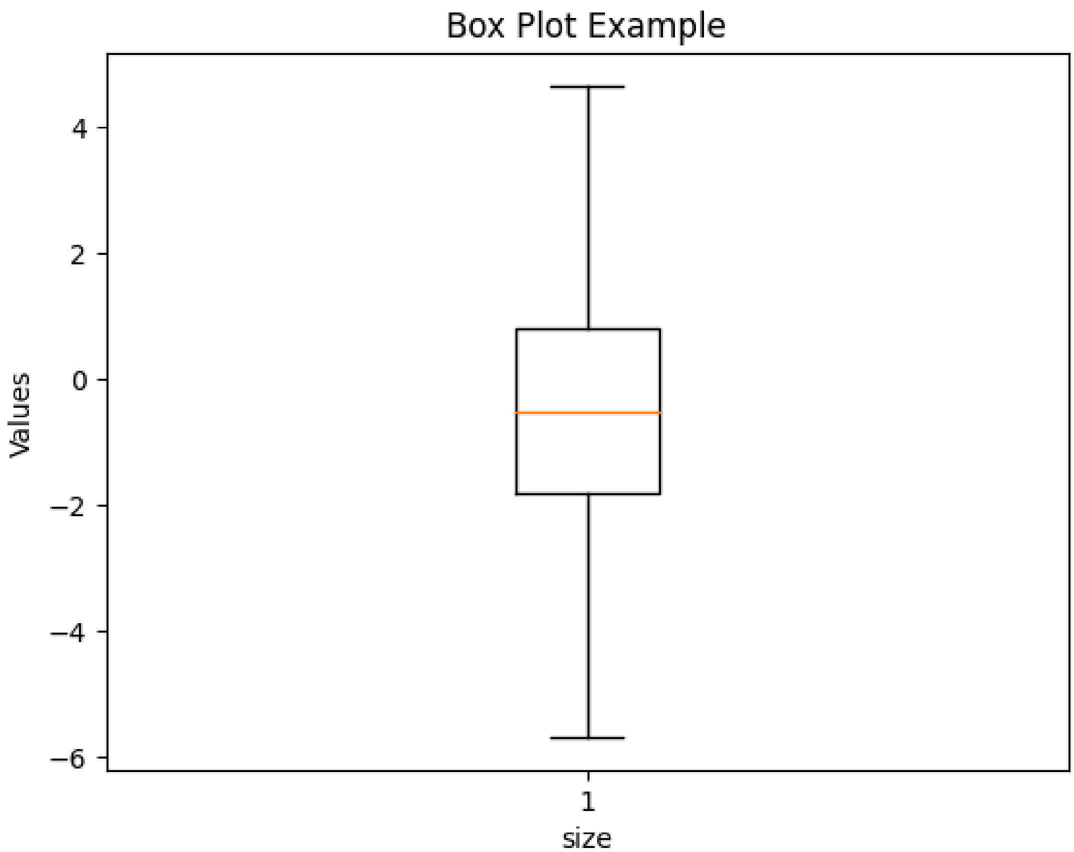



```
filter_data=df_apple.iloc[:,1:7]
filter_data.head()
```

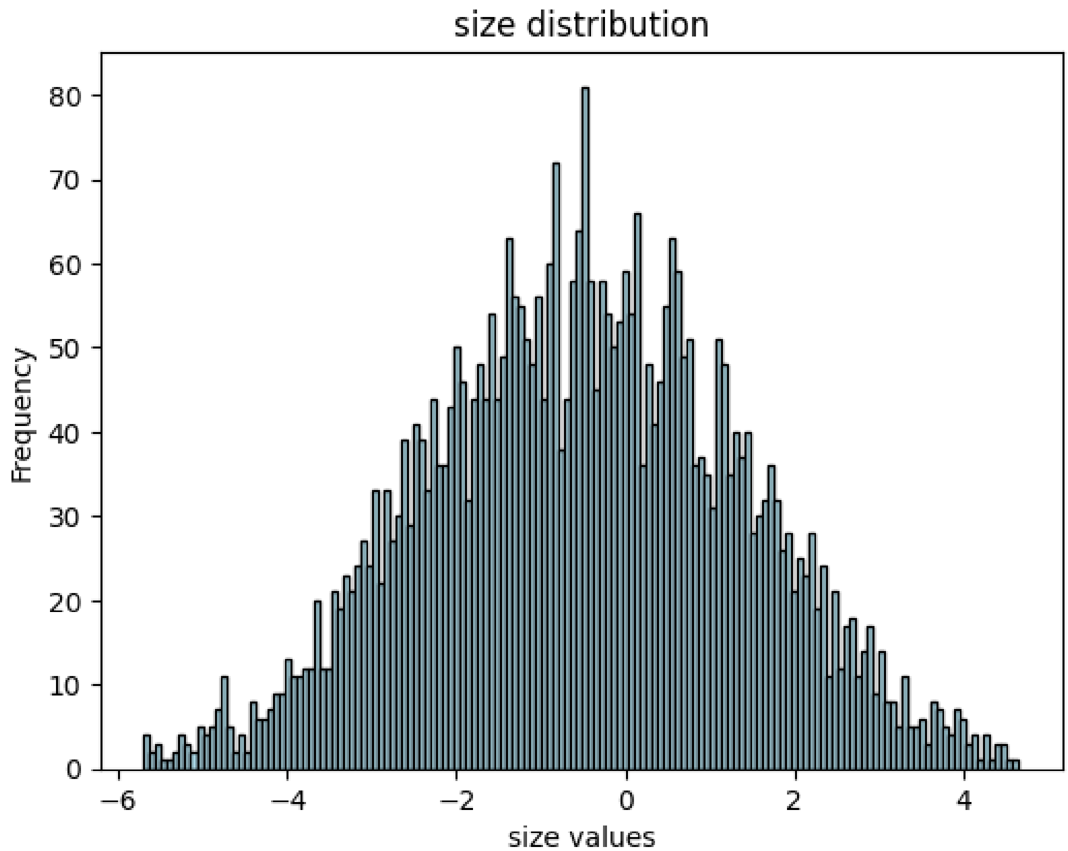
	Weight	Sweetness	Crunchiness	Juiciness	Ripeness	Acidity
0	-2.512336	-0.504758	-1.012009	1.844900	0.329840	-0.491590483
1	-2.839257	3.664059	1.588232	0.853286	0.867530	-0.722809367
2	-1.351282	-1.738429	-0.342616	2.838636	-0.038033	2.621636473
3	-2.271627	1.324874	-0.097875	3.637970	-3.413761	0.790723217
4	-1.296612	-0.384658	-0.553006	3.030874	-1.303849	0.501984036

```
for column in filter_data.columns:
    Q1 = filter_data[column].quantile(0.25)
    Q3 = filter_data[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    mask = (filter_data[column] < lower_bound) | (filter_data[column] >
upper_bound)
    median_value = filter_data[column].median()
    df_apple[column][mask] = median_value

df_apple.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 4000 entries, 0 to 3999
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   A_id         4000 non-null   float64
 1   Size         4000 non-null   float64
 2   Weight       4000 non-null   float64
 3   Sweetness    4000 non-null   float64
 4   Crunchiness  4000 non-null   float64
 5   Juiciness    4000 non-null   float64
 6   Ripeness     4000 non-null   float64
 7   Acidity      4000 non-null   object
 8   Quality      4000 non-null   object
dtypes: float64(7), object(2)
memory usage: 312.5+ KB

df_apple.drop('A_id', axis=1, inplace=True)

df_apple.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 4000 entries, 0 to 3999
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Size         4000 non-null   float64
 1   Weight       4000 non-null   float64
 2   Sweetness    4000 non-null   float64
 3   Crunchiness  4000 non-null   float64
 4   Juiciness    4000 non-null   float64
 5   Ripeness     4000 non-null   float64
 6   Acidity      4000 non-null   object
 7   Quality      4000 non-null   object
dtypes: float64(6), object(2)
memory usage: 410.3+ KB
```

```
counter=df_apple.Size.value_counts()
plt.boxplot(counter.index)

plt.xlabel('size')
plt.ylabel('Values')
plt.title('Box Plot Example')

plt.show()
```




```
plt.hist(df_apple['Size'], bins=150, color='lightblue', edgecolor='black')
plt.xlabel('size values')
plt.ylabel('Frequency')
plt.title('size distribution')

plt.show()
```




```
df_apple.iloc[:,[0,1,2,3,4,5]]
```

	Size	Weight	Sweetness	Crunchiness	Juiciness	Ripeness
0	-3.970049	-2.512336	-0.504758	-1.012009	1.844900	0.329840
1	-1.195217	-2.839257	3.664059	1.588232	0.853286	0.867530
2	-0.292024	-1.351282	-1.738429	-0.342616	2.838636	-0.038033
3	-0.657196	-2.271627	1.324874	-0.097875	3.637970	-3.413761
4	1.364217	-1.296612	-0.384658	-0.553006	3.030874	-1.303849
...	...	...	...	...	...	...
3995	0.059386	-1.067408	-3.714549	0.473052	1.697986	2.244055
3996	-0.293118	1.949253	-0.204020	-0.640196	0.024523	-1.087900
3997	-2.634515	-2.138247	-2.440461	0.657223	2.199709	4.763859
3998	-4.008004	-1.779337	2.366397	-0.200329	2.161435	0.214488
3999	0.278540	-1.715505	0.121217	-1.154075	1.266677	-0.776571

4000 rows × 6 columns

```
correalation=df_apple.iloc[:,[0,1,2,3,4,5]]
correalation
```

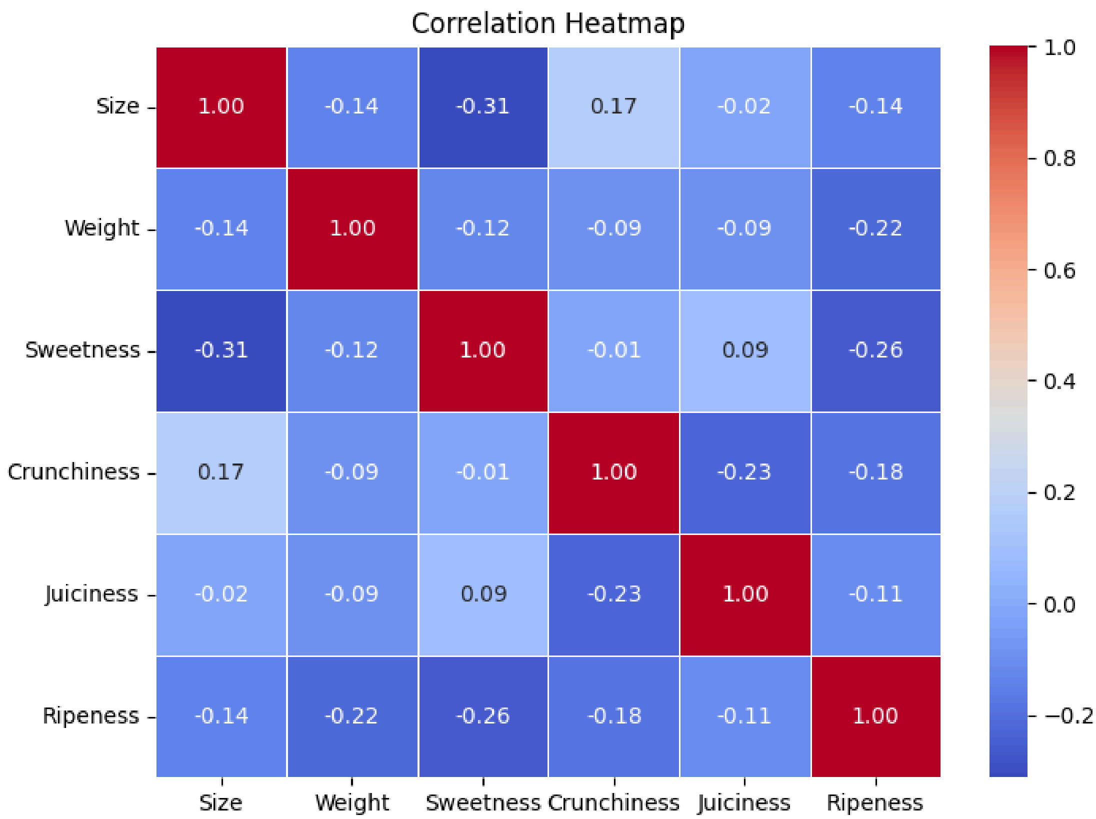
	Size	Weight	Sweetness	Crunchiness	Juiciness	Ripeness
0	-3.970049	-2.512336	-0.504758	-1.012009	1.844900	0.329840
1	-1.195217	-2.839257	3.664059	1.588232	0.853286	0.867530
2	-0.292024	-1.351282	-1.738429	-0.342616	2.838636	-0.038033
3	-0.657196	-2.271627	1.324874	-0.097875	3.637970	-3.413761
4	1.364217	-1.296612	-0.384658	-0.553006	3.030874	-1.303849
...	...	...	...	...	...	...
3995	0.059386	-1.067408	-3.714549	0.473052	1.697986	2.244055
3996	-0.293118	1.949253	-0.204020	-0.640196	0.024523	-1.087900
3997	-2.634515	-2.138247	-2.440461	0.657223	2.199709	4.763859
3998	-4.008004	-1.779337	2.366397	-0.200329	2.161435	0.214488
3999	0.278540	-1.715505	0.121217	-1.154075	1.266677	-0.776571

4000 rows × 6 columns

```
correalation.corr()
```

	Size	Weight	Sweetness	Crunchiness	Juiciness	Ripeness
Size	1.000000	-0.140180	-0.312955	0.165760	-0.022888	-0.139821
Weight	-0.140180	1.000000	-0.120500	-0.086807	-0.090456	-0.221947
Sweetness	-0.312955	-0.120500	1.000000	-0.014191	0.089395	-0.258363
Crunchiness	0.165760	-0.086807	-0.014191	1.000000	-0.227767	-0.181666
Juiciness	-0.022888	-0.090456	0.089395	-0.227767	1.000000	-0.108158
Ripeness	-0.139821	-0.221947	-0.258363	-0.181666	-0.108158	1.000000

```
import seaborn as sns

realtion=correalation.corr()
plt.figure(figsize=(8, 6))
sns.heatmap(realtion ,annot=True, cmap='coolwarm', fmt=".2f", linewidths=.5)
plt.title('Correlation Heatmap')
plt.show()
```


Correlation Heatmap
Size
Weight
Sweetness
Crunchiness
Juiciness
Ripeness
1.00 -0.14 -0.31 0.17 -0.02 -0.14
-0.14 1.00 -0.12 -0.09 -0.09 -0.22
-0.31 -0.12 1.00 -0.01 0.09 -0.26
0.17 -0.09 -0.01 1.00 -0.23 -0.18
-0.02 -0.09 0.09 -0.23 1.00 -0.11
-0.14 -0.22 -0.26 -0.18 -0.11 1.00
1.0
0.8
0.6
0.4
0.2
0.0
−0.2