Exploratory data analysis

Coffee bean quality dataset

Contents

Introduction

The dataset used in this project is based on the Coffee Quality Institute (CQI) Coffee Quality Database and contains 1339 Arabica coffee bean reviews performed by the Coffee Quality Institute, it seeks to predict the overall quality score based on intrinsic features of the coffee beans (such as color, variety, defects), geographical features and information related to the producer, exporter, or grower of the beans**.**

Aim of the Project

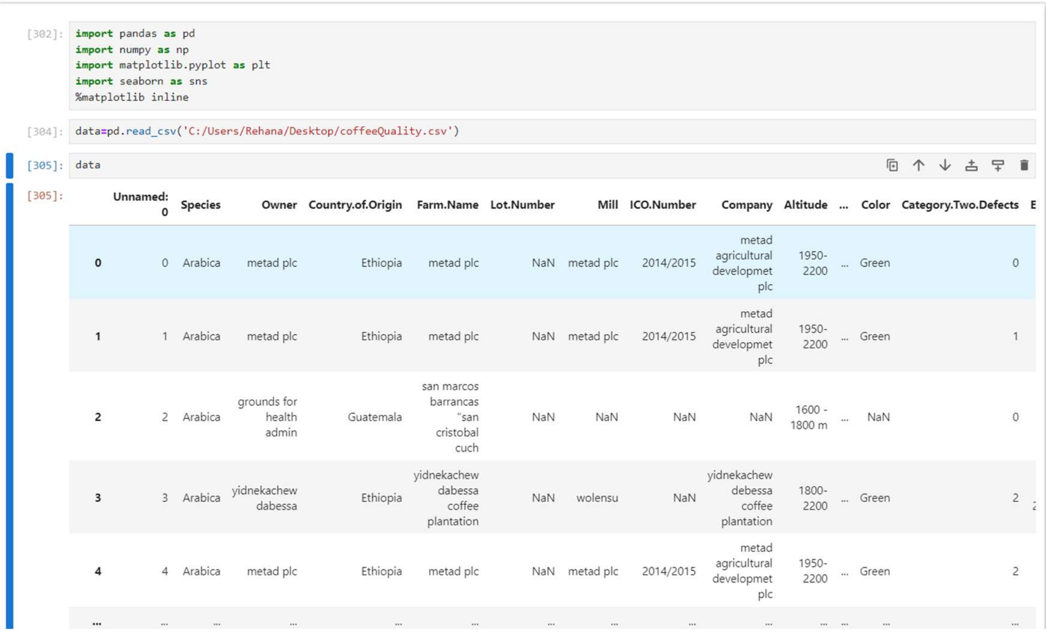
- The primary objective of this project is to conduct a comprehensive analysis to understand the factors that contribute to Coffee quality and improve overall Quality of Coffee .
- By systematically evaluating various parameters such as flavor, aroma, Body, consistency etc., the project aims to identify key factors influencing coffee quality.
- This analysis will provide insights into optimizing Variety of Coffees, Processing Methods etc., and meeting consumer expectations more effectively.

Project Workflow:

Data Understanding

Data preparation is the process of preparing raw data so that it is suitable for further processing and analysis. Key steps include collecting, cleaning, and labeling raw data into a form suitable for further process and then exploring and visualizing the data.

The given coffe quality dataset has been read in the jupyter notebook



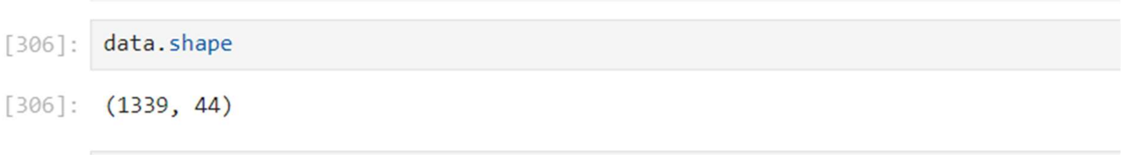
```
[302]: import pandas as pd
       import numpy as np
       import matplotlib.pyplot as plt
       import seaborn as sns
       %matplotlib inline

[304]: data=pd.read_csv('C:/Users/Rehana/Desktop/coffeeQuality.csv')

[305]: data
```

[305]:

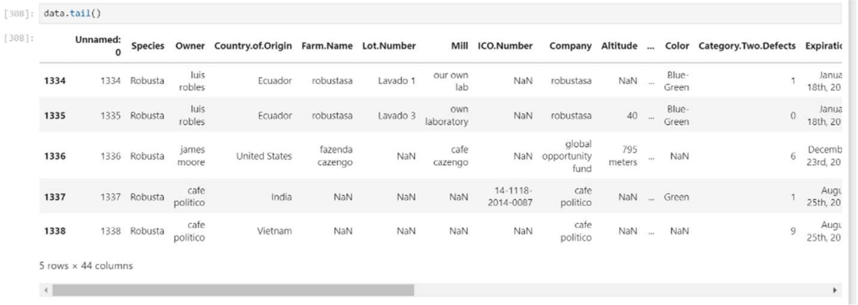
	Unnamed: 0	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	Company	Altitude	...	Color	Category.Two.Defects	E
0	0	Arabica	metad plc	Ethiopia	metad plc	NaN	metad plc	2014/2015	metad agricultural developmet plc	1950-2200	...	Green	0	
1	1	Arabica	metad plc	Ethiopia	metad plc	NaN	metad plc	2014/2015	metad agricultural developmet plc	1950-2200	...	Green	1	
2	2	Arabica	grounds for health admin	Guatemala	san marcos barrancas "san cristobal cuch	NaN	NaN	NaN	NaN	1600 - 1800 m	...	NaN	0	
3	3	Arabica	yidnekachew dabessa	Ethiopia	yidnekachew dabessa coffee plantation	NaN	wolensu	NaN	yidnekachew debessa coffee plantation	1800-2200	...	Green	2	
4	4	Arabica	metad plc	Ethiopia	metad plc	NaN	metad plc	2014/2015	metad agricultural developmet plc	1950-2200	...	Green	2	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	



```
[306]: data.shape

[306]: (1339, 44)
```

The above code is used to define the rows and columns in the given dataset.

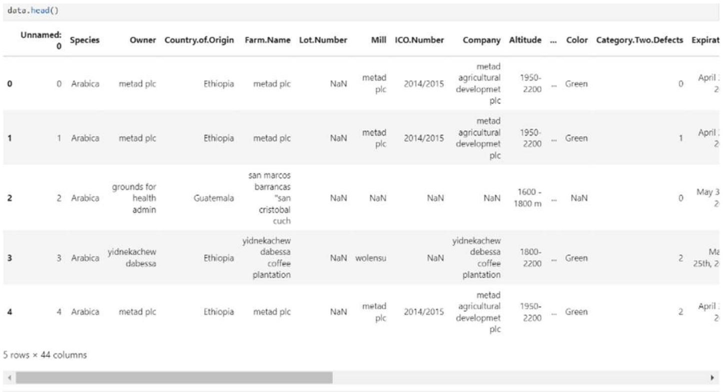


```
[308]: data.tail()
```

[308]:

	Unnamed: 0	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	Company	Altitude	...	Color	Category.Two.Defects	Expiratic
1334	1334	Robusta	luis robles	Ecuador	robustasa	Lavado 1	our own lab	NaN	robustasa	NaN	...	Blue-Green	1	Janua 18th, 20
1335	1335	Robusta	luis robles	Ecuador	robustasa	Lavado 3	own laboratory	NaN	robustasa	40	...	Blue-Green	0	Janua 18th, 20
1336	1336	Robusta	james moore	United States	fazenda cazengo	NaN	cafe cazengo	NaN	global opportunity fund	795 meters	...	NaN	6	Decemb 23rd, 20
1337	1337	Robusta	cafe politico	India	NaN	NaN	NaN	14-1118-2014-0087	cafe politico	NaN	...	Green	1	Augu 25th, 20
1338	1338	Robusta	cafe politico	Vietnam	NaN	NaN	NaN	NaN	cafe politico	NaN	...	NaN	9	Augu 25th, 20

5 rows × 44 columns

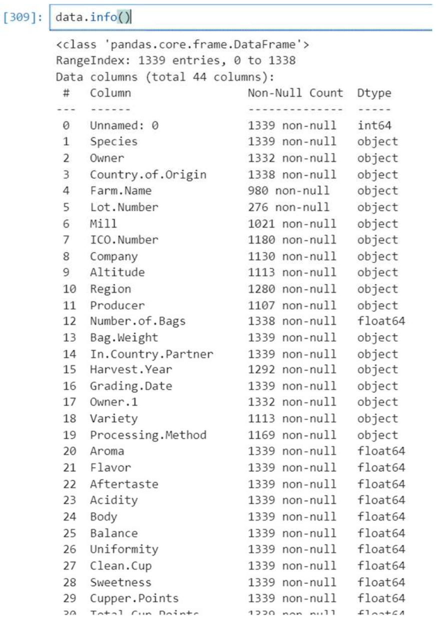


```
data.head()
```

	Unnamed: 0	Species	Owner	Country.of.Origin	Farm.Name	Lot.Number	Mill	ICO.Number	Company	Altitude	...	Color	Category.Two.Defects	Expirat
0	0	Arabica	metad plc	Ethiopia	metad plc	NaN	metad plc	2014/2015	metad agricultural developmet plc	1950-2200	...	Green	0	April 2
1	1	Arabica	metad plc	Ethiopia	metad plc	NaN	metad plc	2014/2015	metad agricultural developmet plc	1950-2200	...	Green	1	April 2
2	2	Arabica	grounds for health admin	Guatemala	san marcos barrancas "san cristobal cuch	NaN	NaN	NaN	NaN	1600 - 1800 m	...	NaN	0	May 3 2
3	3	Arabica	yidnekachew dabessa	Ethiopia	yidnekachew dabessa coffee plantation	NaN	wolensu	NaN	yidnekachew debessa coffee plantation	1800-2200	...	Green	2	Ma 25th, 2
4	4	Arabica	metad plc	Ethiopia	metad plc	NaN	metad plc	2014/2015	metad agricultural developmet plc	1950-2200	...	Green	2	April 2

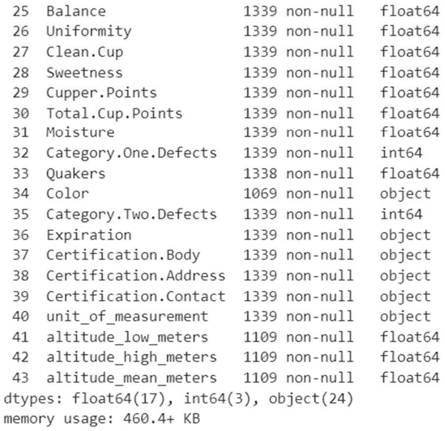
5 rows × 44 columns

The info() method prints information about the DataFrame.The information contains the number of columns, column labels, column data types, memory usage, range index, and the number of cells in each column (non-null values).



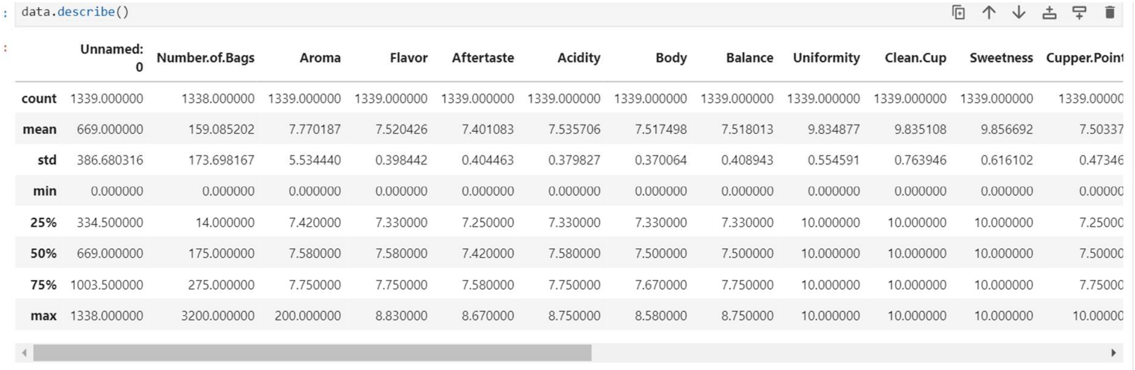
```
[309]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1339 entries, 0 to 1338
Data columns (total 44 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Unnamed: 0            1339 non-null   int64
 1   Species               1339 non-null   object
 2   Owner                 1332 non-null   object
 3   Country.of.Origin     1338 non-null   object
 4   Farm.Name             980 non-null    object
 5   Lot.Number            276 non-null    object
 6   Mill                  1021 non-null   object
 7   ICO.Number            1180 non-null   object
 8   Company               1130 non-null   object
 9   Altitude              1113 non-null   object
 10  Region                1280 non-null   object
 11  Producer              1107 non-null   object
 12  Number.of.Bags        1338 non-null   float64
 13  Bag.Weight            1339 non-null   object
 14  In.Country.Partner    1339 non-null   object
 15  Harvest.Year          1292 non-null   object
 16  Grading.Date          1339 non-null   object
 17  Owner.1               1332 non-null   object
 18  Variety               1113 non-null   object
 19  Processing.Method     1169 non-null   object
 20  Aroma                 1339 non-null   float64
 21  Flavor                1339 non-null   float64
 22  Aftertaste            1339 non-null   float64
 23  Acidity               1339 non-null   float64
 24  Body                  1339 non-null   float64
 25  Balance               1339 non-null   float64
 26  Uniformity            1339 non-null   float64
 27  Clean.Cup             1339 non-null   float64
 28  Sweetness             1339 non-null   float64
 29  Cupper.Points         1339 non-null   float64
 30  Total.Cup.Points      1339 non-null   float64
```



```
 25  Balance               1339 non-null   float64
 26  Uniformity            1339 non-null   float64
 27  Clean.Cup             1339 non-null   float64
 28  Sweetness             1339 non-null   float64
 29  Cupper.Points         1339 non-null   float64
 30  Total.Cup.Points      1339 non-null   float64
 31  Moisture              1339 non-null   float64
 32  Category.One.Defects  1339 non-null   int64
 33  Quakers               1338 non-null   float64
 34  Color                 1069 non-null   object
 35  Category.Two.Defects  1339 non-null   int64
 36  Expiration            1339 non-null   object
 37  Certification.Body    1339 non-null   object
 38  Certification.Address 1339 non-null   object
 39  Certification.Contact 1339 non-null   object
 40  unit_of_measurement   1339 non-null   object
 41  altitude_low_meters   1109 non-null   float64
 42  altitude_high_meters  1109 non-null   float64
 43  altitude_mean_meters  1109 non-null   float64
dtypes: float64(17), int64(3), object(24)
memory usage: 460.4+ KB
```

Data.describe() is used to view some basic statistical details like percentile, mean, std, etc. of a data frame or a series of numeric values. The output is shown in the examples below.



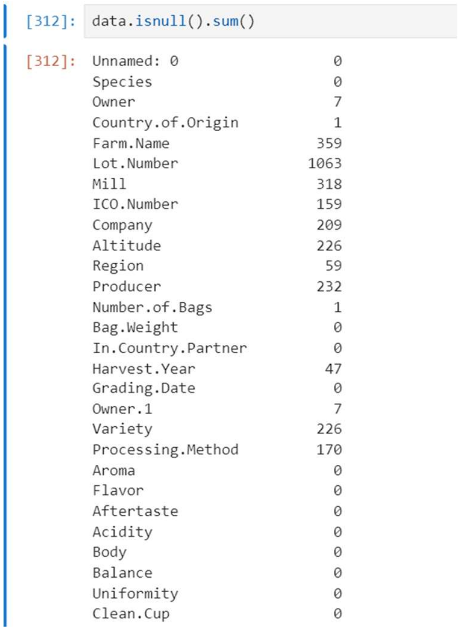
```
data.describe()
```

	Unnamed: 0	Number.of.Bags	Aroma	Flavor	Aftertaste	Acidity	Body	Balance	Uniformity	Clean.Cup	Sweetness	Cupper.Point
count	1339.000000	1338.000000	1339.000000	1339.000000	1339.000000	1339.000000	1339.000000	1339.000000	1339.000000	1339.000000	1339.000000	1339.00000
mean	669.000000	159.085202	7.770187	7.520426	7.401083	7.535706	7.517498	7.518013	9.834877	9.835108	9.856692	7.50337
std	386.680316	173.698167	5.534440	0.398442	0.404463	0.379827	0.370064	0.408943	0.554591	0.763946	0.616102	0.47346
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
25%	334.500000	14.000000	7.420000	7.330000	7.250000	7.330000	7.330000	7.330000	10.000000	10.000000	10.000000	7.25000
50%	669.000000	175.000000	7.580000	7.580000	7.420000	7.580000	7.500000	7.500000	10.000000	10.000000	10.000000	7.50000
75%	1003.500000	275.000000	7.750000	7.750000	7.580000	7.750000	7.670000	7.750000	10.000000	10.000000	10.000000	7.75000
max	1338.000000	3200.000000	200.000000	8.830000	8.670000	8.750000	8.580000	8.750000	10.000000	10.000000	10.000000	10.00000

Data Cleaning

Data cleaning involves revising, rectifying, and organizing information in a dataset to make it consistent and ready for analysis. This step entails identifying and addressing errors, inconsistencies, duplicates, or incomplete entries within the data. The main objective of data cleaning is to enhance the data's quality and usefulness, thereby leading to more dependable and precise findings.

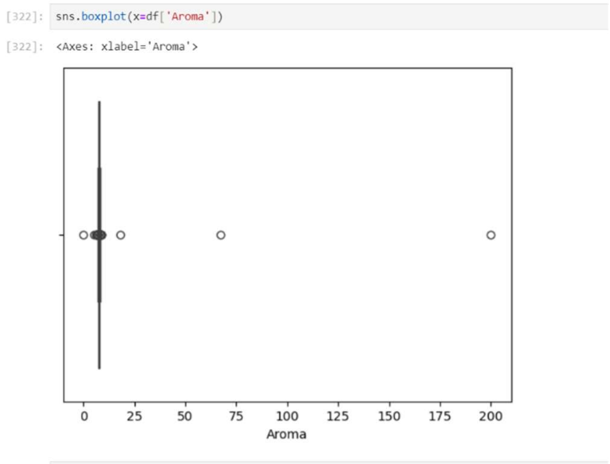
Data. isnull(). sum()returns the number of missing values in the dataset.



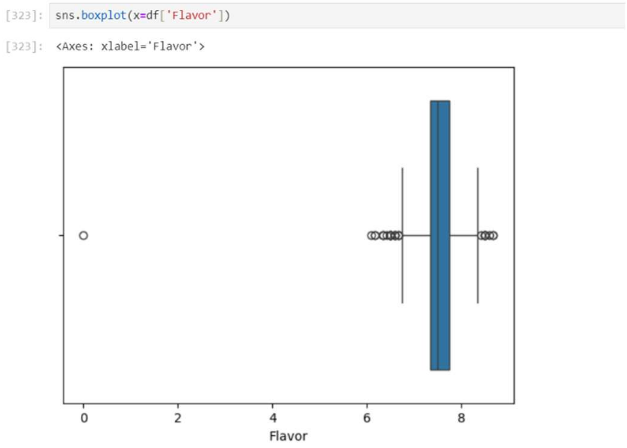
```
[312]: data.isnull().sum()
```

```
[312]: Unnamed: 0                0
       Species                   0
       Owner                     7
       Country.of.Origin         1
       Farm.Name               359
       Lot.Number             1063
       Mill                    318
       ICO.Number              159
       Company                 209
       Altitude                226
       Region                   59
       Producer                232
       Number.of.Bags            1
       Bag.Weight                0
       In.Country.Partner        0
       Harvest.Year             47
       Grading.Date              0
       Owner.1                   7
       Variety                 226
       Processing.Method       170
       Aroma                     0
       Flavor                    0
       Aftertaste                0
       Acidity                   0
       Body                      0
       Balance                   0
       Uniformity                0
       Clean.Cup                 0
```

- After find the missings values they can be replaced by find the mean, median if the columns are numerical in nature.
- And mode can be applied to the categorical data.
- Outliers can be visually seen by using the boxplot and scatter plot



```
[322]: sns.boxplot(x=df['Aroma'])
[322]: <Axes: xlabel='Aroma'>
```

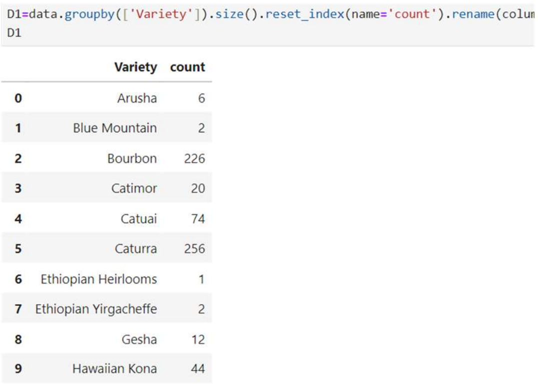


```
[323]: sns.boxplot(x=df['Flavor'])
[323]: <Axes: xlabel='Flavor'>
```

Data filtering

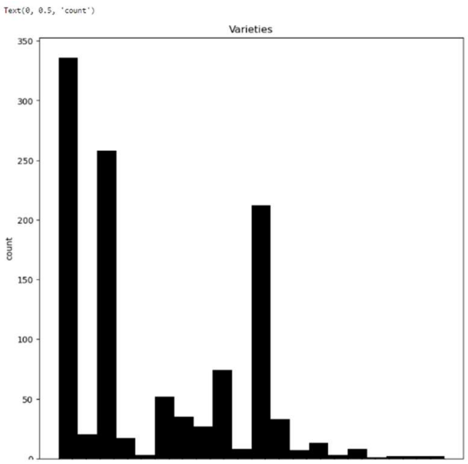
Filtering data means choosing or not choosing certain information from a set of data using a set of criteria. This is important for finding important data, getting rid of unnecessary information, and improving the overall quality of the data.

Dropna.(),groupby(), etc can be used to get particular colums required to be used for further analysis.



```
D1=data.groupby(['Variety']).size().reset_index(name='count').rename(colum
D1
```

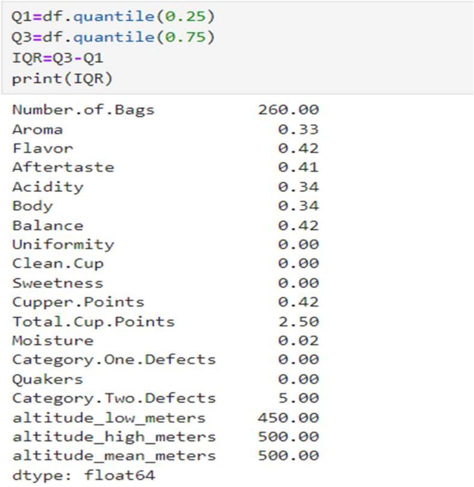
	Variety	count
0	Arusha	6
1	Blue Mountain	2
2	Bourbon	226
3	Catimor	20
4	Catuai	74
5	Caturra	256
6	Ethiopian Heirlooms	1
7	Ethiopian Yirgacheffe	2
8	Gesha	12
9	Hawaiian Kona	44



```
Text(0, 0.5, 'count')
```

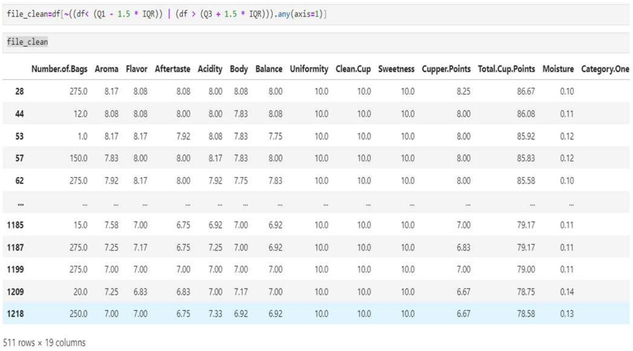
The quantile () method calculates the quantile of the values in a given axis. Default axis is row.

By specifying the column axis (axis='columns'), the quantile () method calculates the quantile column-wise and returns the mean value for each *row*.



```
Q1=df.quantile(0.25)
Q3=df.quantile(0.75)
IQR=Q3-Q1
print(IQR)
```

```
Number.of.Bags            260.00
Aroma                       0.33
Flavor                      0.42
Aftertaste                  0.41
Acidity                     0.34
Body                        0.34
Balance                     0.42
Uniformity                  0.00
Clean.Cup                   0.00
Sweetness                   0.00
Cupper.Points               0.42
Total.Cup.Points            2.50
Moisture                    0.02
Category.One.Defects        0.00
Quakers                     0.00
Category.Two.Defects        5.00
altitude_low_meters       450.00
altitude_high_meters      500.00
altitude_mean_meters      500.00
dtype: float64
```



```
file_clean=df[~((df< (Q1 - 1.5 * IQR)) | (df > (Q3 + 1.5 * IQR))).any(axis=1)]
```

```
file_clean
```

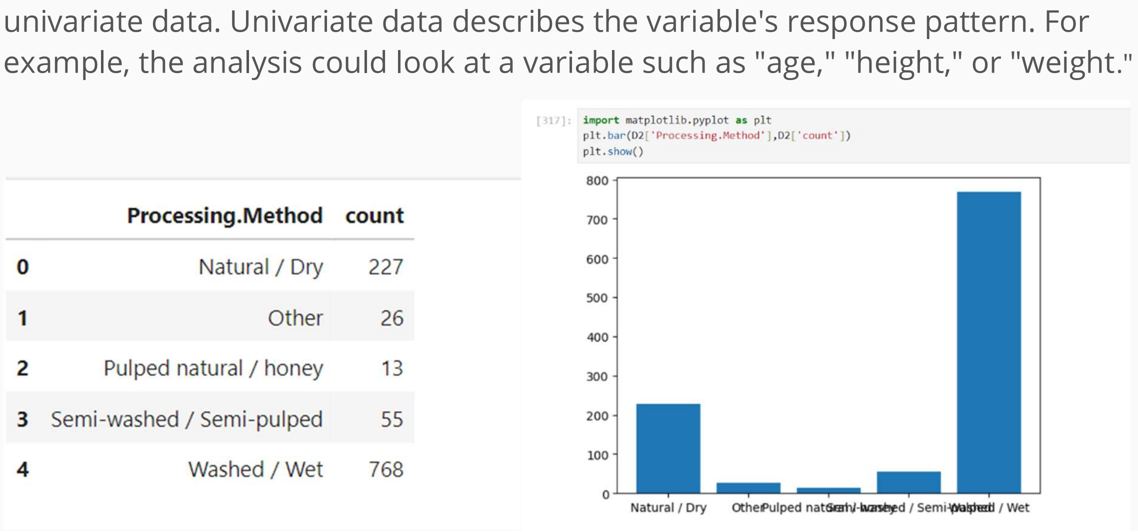
	Number.of.Bags	Aroma	Flavor	Aftertaste	Acidity	Body	Balance	Uniformity	Clean.Cup	Sweetness	Cupper.Points	Total.Cup.Points	Moisture	Category.One
28	275.0	8.17	8.08	8.08	8.00	8.08	8.00	10.0	10.0	10.0	8.25	86.67	0.10	
44	12.0	8.08	8.08	8.00	8.00	7.83	8.08	10.0	10.0	10.0	8.00	86.08	0.11	
53	1.0	8.17	8.17	7.92	8.08	7.83	7.75	10.0	10.0	10.0	8.00	85.92	0.12	
57	150.0	7.83	8.00	8.00	8.17	7.83	8.00	10.0	10.0	10.0	8.00	85.83	0.12	
62	275.0	7.92	8.17	8.00	7.92	7.75	7.83	10.0	10.0	10.0	8.00	85.58	0.10	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1185	15.0	7.58	7.00	6.75	6.92	7.00	6.92	10.0	10.0	10.0	7.00	79.17	0.11	
1187	275.0	7.25	7.17	6.75	7.25	7.00	6.92	10.0	10.0	10.0	6.83	79.17	0.11	
1199	275.0	7.00	7.00	7.00	7.00	7.00	7.00	10.0	10.0	10.0	7.00	79.00	0.11	
1209	20.0	7.25	6.83	6.83	7.00	7.17	7.00	10.0	10.0	10.0	6.67	78.75	0.14	
1218	250.0	7.00	7.00	6.75	7.33	6.92	6.92	10.0	10.0	10.0	6.67	78.58	0.13	

511 rows × 19 columns

After filtering and cleaning the data it is now ready to do further analysis needed .

Univariate analysis

Univariate data is a term used in statistics to describe data that consists of observations on only one characteristic or attribute. There is only one variable in



univariate data. Univariate data describes the variable's response pattern. For example, the analysis could look at a variable such as "age," "height," or "weight."

	Processing.Method	count
0	Natural / Dry	227
1	Other	26
2	Pulped natural / honey	13
3	Semi-washed / Semi-pulped	55
4	Washed / Wet	768

```
[317]: import matplotlib.pyplot as plt
       plt.bar(D2['Processing.Method'],D2['count'])
       plt.show()
```

Bivariate analysis

Bivariate analysis is a statistical method to determine if there is a statistical link between the two variables and, if so, how strong and in which direction that link is.It is a helpful technique for determining how two variables are connected and finding trends and patterns in the data. In statistical analysis, distinguishing between categorical data and numerical data is essential, as categorical data involves distinct categories or labels, while numerical data consists of measurable quantities.

Examples of bivariate analysis

Scatterplots

Correlation

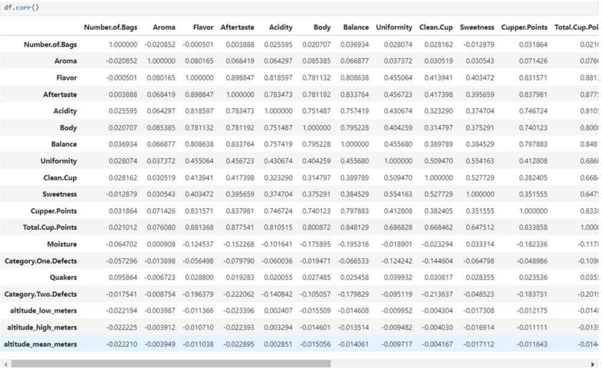
Regression

Chi square

T test and ANOVA(Analysis of variance)

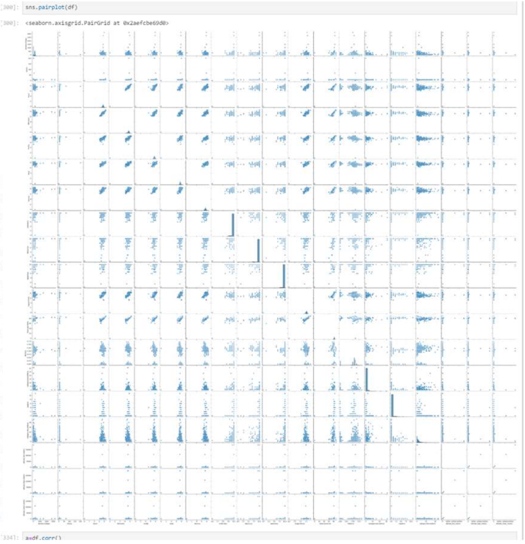
Correlation

Correlation is a statistical measure that shows how strong and in what direction two variables are linked. A positive correlation means that when one variable goes up, so does the other. A negative correlation shows that when one variable goes up, the other one goes down.



```
df.corr()
```

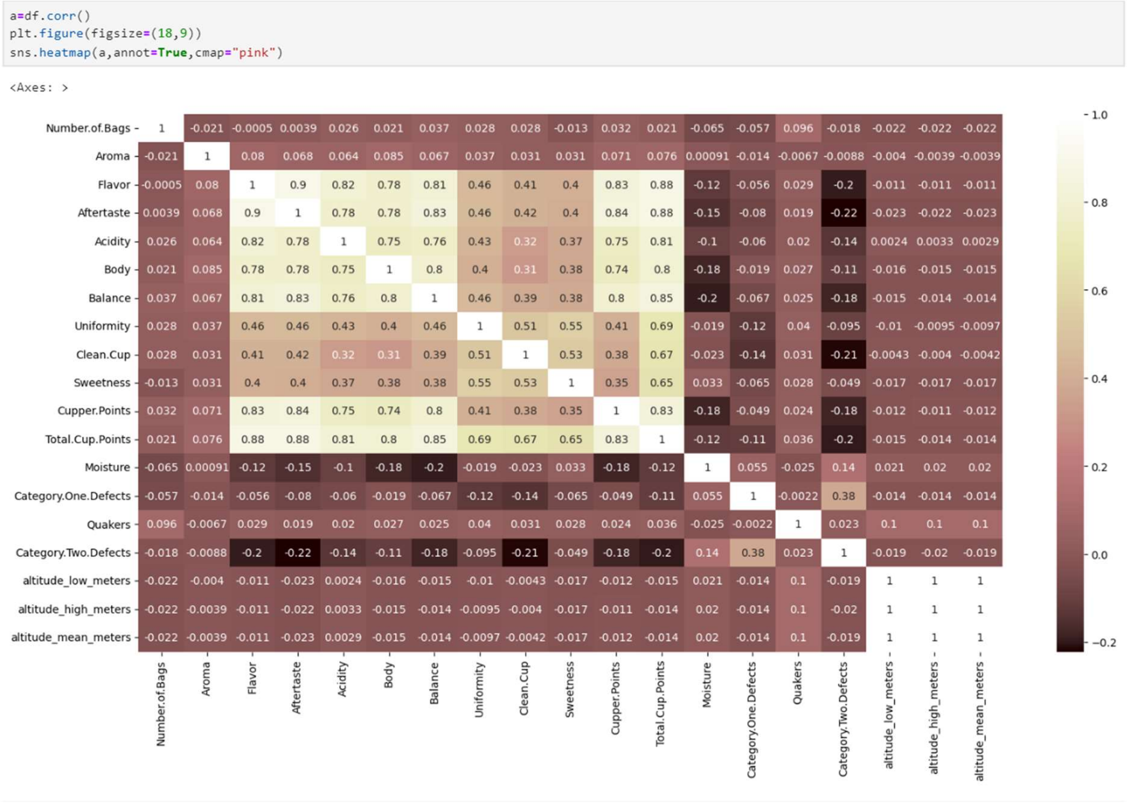
	Number.of.Bags	Aroma	Flavor	Aftertaste	Acidity	Body	Balance	Uniformity	Clean.Cup	Sweetness	Cupper.Points	Total.Cup.Poi
Number.of.Bags	1.000000	-0.020852	-0.000501	0.003888	0.025595	0.020707	0.036934	0.028074	0.028162	-0.012879	0.031864	0.021
Aroma	-0.020852	1.000000	0.080165	0.068419	0.064297	0.085385	0.066877	0.037372	0.030519	0.030543	0.071426	0.076
Flavor	-0.000501	0.080165	1.000000	0.898847	0.818597	0.781132	0.808638	0.455064	0.413941	0.403472	0.831571	0.881
Aftertaste	0.003888	0.068419	0.898847	1.000000	0.783473	0.781192	0.833764	0.456723	0.417398	0.395659	0.837981	0.877
Acidity	0.025595	0.064297	0.818597	0.783473	1.000000	0.751487	0.757419	0.430674	0.323290	0.374704	0.746724	0.810
Body	0.020707	0.085385	0.781132	0.781192	0.751487	1.000000	0.795228	0.404259	0.314797	0.375291	0.740123	0.800
Balance	0.036934	0.066877	0.808638	0.833764	0.757419	0.795228	1.000000	0.455680	0.389789	0.384529	0.797883	0.848
Uniformity	0.028074	0.037372	0.455064	0.456723	0.430674	0.404259	0.455680	1.000000	0.509470	0.554163	0.412808	0.686
Clean.Cup	0.028162	0.030519	0.413941	0.417398	0.323290	0.314797	0.389789	0.509470	1.000000	0.527729	0.382405	0.668
Sweetness	-0.012879	0.030543	0.403472	0.395659	0.374704	0.375291	0.384529	0.554163	0.527729	1.000000	0.351555	0.647
Cupper.Points	0.031864	0.071426	0.831571	0.837981	0.746724	0.740123	0.797883	0.412808	0.382405	0.351555	1.000000	0.833
Total.Cup.Points	0.021012	0.076080	0.881368	0.877541	0.810515	0.800872	0.848129	0.686828	0.668462	0.647512	0.833858	1.000
Moisture	-0.064702	0.000908	-0.124537	-0.152268	-0.101641	-0.175895	-0.195316	-0.018901	-0.023294	0.033314	-0.182336	-0.117
Category.One.Defects	-0.057296	-0.013898	-0.056498	-0.079790	-0.060036	-0.019471	-0.066533	-0.124242	-0.144604	-0.064798	-0.048986	-0.109
Quakers	0.095864	-0.006723	0.028800	0.019283	0.020055	0.027485	0.025458	0.039932	0.030817	0.028355	0.023536	0.035
Category.Two.Defects	-0.017541	-0.008754	-0.196379	-0.222062	-0.140842	-0.105057	-0.179829	-0.095119	-0.213637	-0.048523	-0.183731	-0.201
altitude_low_meters	-0.022194	-0.003987	-0.011366	-0.023396	0.002407	-0.015509	-0.014608	-0.009952	-0.004304	-0.017308	-0.012175	-0.014
altitude_high_meters	-0.022225	-0.003912	-0.010710	-0.022393	0.003294	-0.014601	-0.013514	-0.009482	-0.004030	-0.016914	-0.011111	-0.013
altitude_mean_meters	-0.022210	-0.003949	-0.011038	-0.022895	0.002851	-0.015056	-0.014061	-0.009717	-0.004167	-0.017112	-0.011643	-0.014



```
[300]: sns.pairplot(df)

[300]: <seaborn.axisgrid.PairGrid at 0x2aefcbe69d0>

[334]: a=df.corr()
```



```
a=df.corr()
plt.figure(figsize=(18,9))
sns.heatmap(a,annot=True,cmap="pink")
```

```
<Axes: >
```

Insight obtained from the Analysis

- Category 2 defects are more compared to category 1 defects.
- Washed/Wet processed beans has more defects.
- November, April & March Graded Beans has more defects.
- Beans with Moisture Range more than 10% has more defects.
- Harvest year 2021/2022 & 2022/2023 has more defected beans.

Note : Here, the Coffee Bean production in the respective years and Processed method is high, so it may also be the reason for more def