ALORRO, JAY-ANN S.

CPE311 - CPE22S3

SEATWORK 6.1

APPLE QUALITY

DATA IMPUTATION/ CLEANING

Initial dataset

	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
...	...	...	...	...	...	...	...	...	...
3996	3996.0	-0.293118	1.949253	-0.204020	-0.640196	0.024523	-1.087900	1.854235285	good
3997	3997.0	-2.634515	-2.138247	-2.440461	0.657223	2.199709	4.763859	-1.334611391	bad
3998	3998.0	-4.008004	-1.779337	2.366397	-0.200329	2.161435	0.214488	-2.229719806	good
3999	3999.0	0.278540	-1.715505	0.121217	-1.154075	1.266677	-0.776571	1.599796456	good
4000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Created_by_Nidula_Elgiriyewithana	NaN

4001 rows × 9 columns

Removing unnecessary data and checking for data duplication.

```
[86] # removing nan values (only 1 which is the last row, the author rights)
     df = data.dropna()
     df = data.replace([np.inf, -np.inf], np.nan).dropna()
```

```
[87] # checking for duplicate rows
     df.duplicated().sum()
```

```
     0
```

Assigning a new index column.

```
[88] # setting data type of A_id as int from float64
     df['A_id'] = df['A_id'].astype(int)
```

```
df
```

	A_id	Size	Weight	Sweetness	Crunchiness	Juiciness	Ripeness	Acidity	Quality
0	0	-3.970049	-2.512336	5.346330	-1.012009	1.844900	0.329840	-0.491590483	good
1	1	-1.195217	-2.839257	3.664059	1.588232	0.853286	0.867530	-0.722809367	good
2	2	-0.292024	-1.351282	-1.738429	-0.342616	2.838636	-0.038033	2.621636473	bad
3	3	-0.657196	-2.271627	1.324874	-0.097875	3.637970	-3.413761	0.790723217	good
4	4	1.364217	-1.296612	-0.384658	-0.553006	3.030874	-1.303849	0.501984036	good
...	...	...	...	...	...	...	...	...	...
3995	3995	0.059386	-1.067408	-3.714549	0.473052	1.697986	2.244055	0.137784369	bad
3996	3996	-0.293118	1.949253	-0.204020	-0.640196	0.024523	-1.087900	1.854235285	good
3997	3997	-2.634515	-2.138247	-2.440461	0.657223	2.199709	4.763859	-1.334611391	bad
3998	3998	-4.008004	-1.779337	2.366397	-0.200329	2.161435	0.214488	-2.229719806	good
3999	3999	0.278540	-1.715505	0.121217	-1.154075	1.266677	-0.776571	1.599796456	good

4000 rows × 9 columns

```
[90] # setting A_id as index
     df.set_index('A_id', inplace=True)
```

```
df
```

A_id	Size	Weight	Sweetness	Crunchiness	Juiciness	Ripeness	Acidity	Quality
0	-3.970049	-2.512336	5.346330	-1.012009	1.844900	0.329840	-0.491590483	good
1	-1.195217	-2.839257	3.664059	1.588232	0.853286	0.867530	-0.722809367	good
2	-0.292024	-1.351282	-1.738429	-0.342616	2.838636	-0.038033	2.621636473	bad
3	-0.657196	-2.271627	1.324874	-0.097875	3.637970	-3.413761	0.790723217	good
4	1.364217	-1.296612	-0.384658	-0.553006	3.030874	-1.303849	0.501984036	good
...	...	...	...	...	...	...	...	...
3995	0.059386	-1.067408	-3.714549	0.473052	1.697986	2.244055	0.137784369	bad
3996	-0.293118	1.949253	-0.204020	-0.640196	0.024523	-1.087900	1.854235285	good
3997	-2.634515	-2.138247	-2.440461	0.657223	2.199709	4.763859	-1.334611391	bad
3998	-4.008004	-1.779337	2.366397	-0.200329	2.161435	0.214488	-2.229719806	good
3999	0.278540	-1.715505	0.121217	-1.154075	1.266677	-0.776571	1.599796456	good

4000 rows × 8 columns

Other details of the dataset.

```
[92] # column names
     df.columns
```

```
Index(['Size', 'Weight', 'Sweetness', 'Crunchiness', 'Juiciness', 'Ripeness',
       'Acidity', 'Quality'],
      dtype='object')
```

```
[93] # types of the data
     df.dtypes
```

```
Size           float64
Weight         float64
Sweetness      float64
Crunchiness    float64
Juiciness      float64
Ripeness       float64
Acidity         object
Quality         object
dtype: object
```

```
[94] # total no of records
     len(df)
```

```
4000
```

Converting one data type to another and creating new column for representation.

```
[95] # converting acidity from object to float
     df['Acidity'] = pd.to_numeric(df['Acidity'], errors='coerce')
```

```
df.dtypes
```

```
Size           float64
Weight         float64
Sweetness      float64
Crunchiness    float64
Juiciness      float64
Ripeness       float64
Acidity        float64
Quality         object
dtype: object
```

```
[97] # if quality is good, grade == 1, else == 0
     df['Grade'] = np.where(df['Quality'] == 'good', 1, 0)
```

A_id	Size	Weight	Sweetness	Crunchiness	Juiciness	Ripeness	Acidity	Quality	Grade
0	-3.970049	-2.512336	5.346330	-1.012009	1.844900	0.329840	-0.491590	good	1
1	-1.195217	-2.839257	3.664059	1.588232	0.853286	0.867530	-0.722809	good	1
2	-0.292024	-1.351282	-1.738429	-0.342616	2.838636	-0.038033	2.621636	bad	0
3	-0.657196	-2.271627	1.324874	-0.097875	3.637970	-3.413761	0.790723	good	1
4	1.364217	-1.296612	-0.384658	-0.553006	3.030874	-1.303849	0.501984	good	1
...	...	...	...	...	...	...	...	...	...
3995	0.059386	-1.067408	-3.714549	0.473052	1.697986	2.244055	0.137784	bad	0
3996	-0.293118	1.949253	-0.204020	-0.640196	0.024523	-1.087900	1.854235	good	1
3997	-2.634515	-2.138247	-2.440461	0.657223	2.199709	4.763859	-1.334611	bad	0
3998	-4.008004	-1.779337	2.366397	-0.200329	2.161435	0.214488	-2.229720	good	1
3999	0.278540	-1.715505	0.121217	-1.154075	1.266677	-0.776571	1.599796	good	1

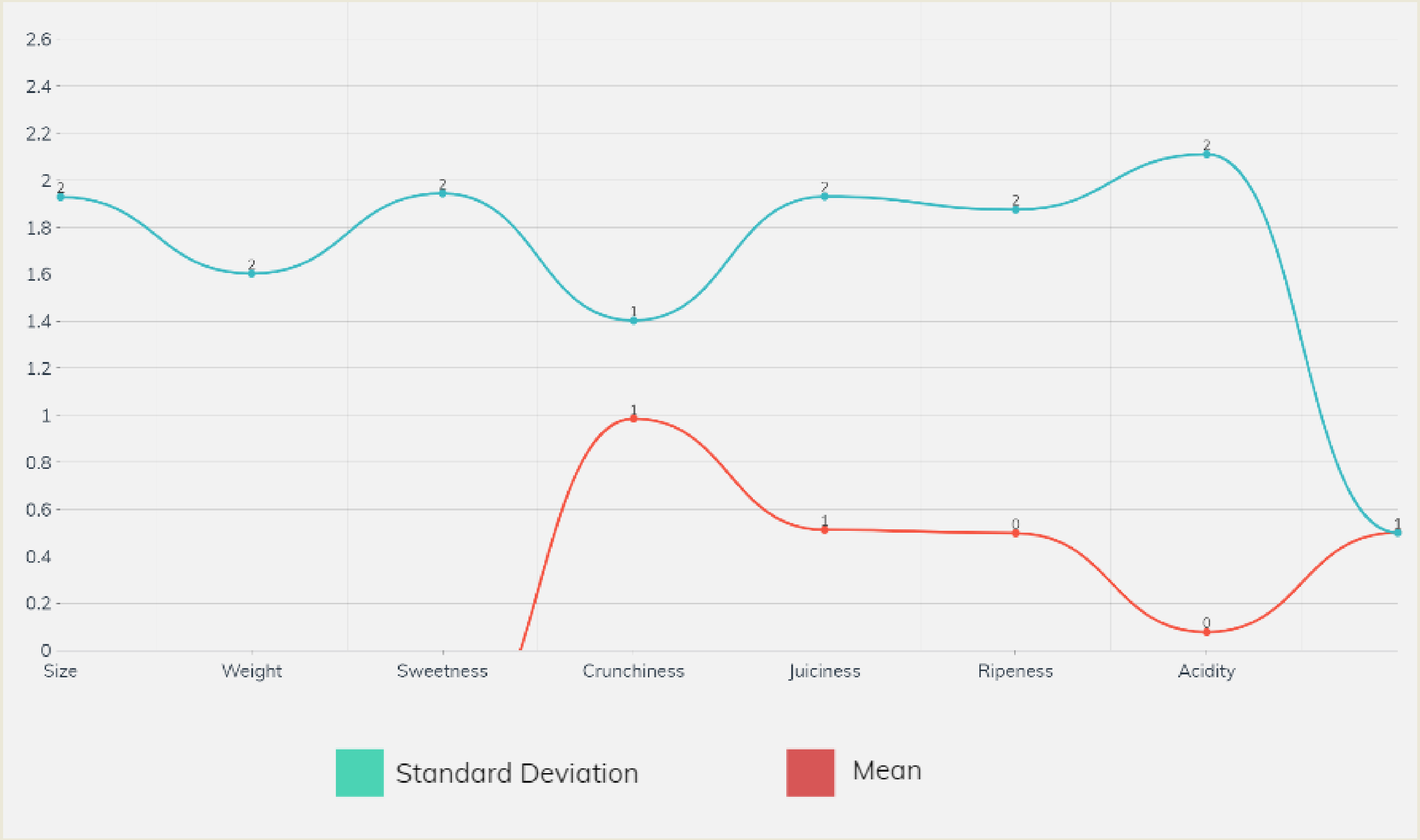
4000 rows × 9 columns

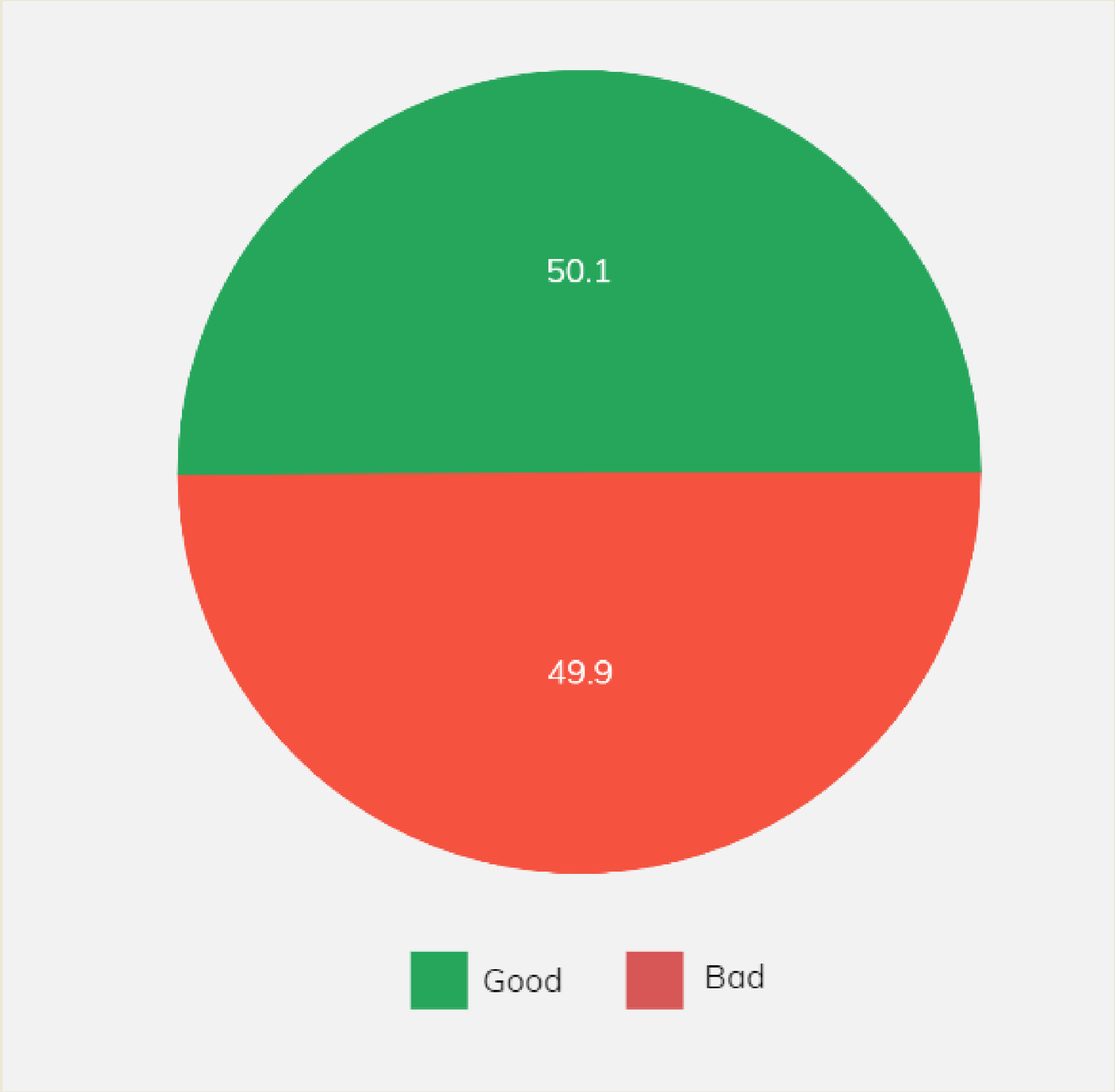
DATA ANALYSIS AND STATISTICS

Statistical Summary

```
[102] # summarization of the data
      df.describe()
```

	Size	Weight	Sweetness	Crunchiness	Juiciness	Ripeness	Acidity	Grade
count	4000.000000	4000.000000	4000.000000	4000.000000	4000.000000	4000.000000	4000.000000	4000.000000
mean	-0.503015	-0.989547	-0.470479	0.985478	0.512118	0.498277	0.076877	0.501000
std	1.928059	1.602507	1.943441	1.402757	1.930286	1.874427	2.110270	0.500062
min	-7.151703	-7.149848	-6.894485	-6.055058	-5.961897	-5.864599	-7.010538	0.000000
25%	-1.816765	-2.011770	-1.738425	0.062764	-0.801286	-0.771677	-1.377424	0.000000
50%	-0.513703	-0.984736	-0.504758	0.998249	0.534219	0.503445	0.022609	1.000000
75%	0.805526	0.030976	0.801922	1.894234	1.835976	1.766212	1.510493	1.000000
max	6.406367	5.790714	6.374916	7.619852	7.364403	7.237837	7.404736	1.000000


2.6
2.4
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
Size
Weight
Sweetness
Crunchiness
Juiciness
Ripeness
Acidity
Standard Deviation
Mean


50.1
49.9
Good
Bad

Mean

```
[99] # mean for the size, weight, sweetness, crunchiness, juiciness, ripeness, acidity, quality (using numpy)
     size_mean = np.mean(df['Size'])
     weight_mean = np.mean(df['Weight'])
     sweetness_mean = np.mean(df['Sweetness'])
     crunchiness_mean = np.mean(df['Crunchiness'])
     juiciness_mean = np.mean(df['Juiciness'])
     ripeness_mean = np.mean(df['Ripeness'])
     acidity_mean = np.mean(df['Acidity'])
     quality_mean = np.mean(df['Grade'])

     print('Mean of Size:', size_mean)
     print('Mean of Weight:', weight_mean)
     print('Mean of Sweetness:', sweetness_mean)
     print('Mean of Crunchiness:', crunchiness_mean)
     print('Mean of Juiciness:', juiciness_mean)
     print('Mean of Ripeness:', ripeness_mean)
     print('Mean of Acidity:', acidity_mean)
     print('Mean of Quality:', quality_mean)
```

```
Mean of Size: -0.50301462982675
Mean of Weight: -0.9895465445945
Mean of Sweetness: -0.47047851978824995
Mean of Crunchiness: 0.9854779038585
Mean of Juiciness: 0.5121179684932501
Mean of Ripeness: 0.4982774280305
Mean of Acidity: 0.07687729571600001
Mean of Quality: 0.501
```

MEAN

Size:

- The mean size of the fruits in the dataset is approximately -0.50.

Weight:

- The mean weight of the fruits is approximately -0.99.

Sweetness:

- The mean sweetness level is approximately -0.47.

Crunchiness:

- The mean crunchiness level is approximately 0.99.

Juiciness:

- The mean juiciness level is approximately 0.51.

Ripeness:

- The mean ripeness level is approximately 0.50.

Acidity:

- The mean acidity level is approximately 0.08.

Quality/Grade:

- The data indicates that the grade variable is categorical, with values of either 0 or 1. The mean grade is approximately 0.50, suggesting a balanced distribution between the two categories.

Standard Deviation

```
[101] # standard deviation for the size, weight, sweetness, crunchiness, juiciness, ripeness, acidity, quality (using numpy)
      size_std = np.std(df['Size'])
      weight_std = np.std(df['Weight'])
      sweetness_std = np.std(df['Sweetness'])
      crunchiness_std = np.std(df['Crunchiness'])
      juiciness_std = np.std(df['Juiciness'])
      ripeness_std = np.std(df['Ripeness'])
      acidity_std = np.std(df['Acidity'])
      quality_std = np.std(df['Grade'])

      print('Standard Deviation of Size:', size_std)
      print('Standard Deviation of Weight:', weight_std)
      print('Standard Deviation of Sweetness:', sweetness_std)
      print('Standard Deviation of Crunchiness:', crunchiness_std)
      print('Standard Deviation of Juiciness:', juiciness_std)
      print('Standard Deviation of Ripeness:', ripeness_std)
      print('Standard Deviation of Acidity:', acidity_std)
      print('Standard Deviation of Quality:', quality_std)
```

```
Standard Deviation of Size: 1.9278176664540305
Standard Deviation of Weight: 1.602306888228833
Standard Deviation of Sweetness: 1.9431977136530587
Standard Deviation of Crunchiness: 1.4025818486010257
Standard Deviation of Juiciness: 1.9300443723029157
Standard Deviation of Ripeness: 1.8741924577105886
Standard Deviation of Acidity: 2.1100058362278236
Standard Deviation of Quality: 0.499998999999
```

STANDARD DEVIATION

Size:

- The standard deviation is approximately 1.93, indicating a considerable variability in fruit sizes.
- The smallest observed size is around -7.15, and the largest observed size is around 6.41.

Weight:

- The standard deviation is approximately 1.60, indicating variability in fruit weights.
- The smallest observed weight is around -7.15, and the largest observed weight is around 5.79.

Sweetness:

- The standard deviation is approximately 1.94, indicating variability in sweetness levels.
- The lowest observed sweetness level is around -6.89, and the highest observed sweetness level is around 6.37.

Crunchiness:

- The standard deviation is approximately 1.40, indicating variability in crunchiness levels.
- The lowest observed crunchiness level is around -6.06, and the highest observed crunchiness level is around 7.62.

Juiciness:

- The standard deviation is approximately 1.93, indicating variability in juiciness levels.
- The lowest observed juiciness level is around -5.96, and the highest observed juiciness level is around 7.36.

Ripeness:

- The standard deviation is approximately 1.87, indicating variability in ripeness levels.
- The lowest observed ripeness level is around -5.86, and the highest observed ripeness level is around 7.24.

Acidity:

- The standard deviation is approximately 2.11, indicating variability in acidity levels.
- The lowest observed acidity level is around -7.01, and the highest observed acidity level is around 7.40.

Quality/Grade:

- The data indicates that the grade variable is categorical, with values of either 0 or 1. The mean grade is approximately 0.50, suggesting a balanced distribution between the two categories.

SUMMARY

In conclusion, the dataset used provides a statistical summary for the different attributes that is related to an apply including size, weight, sweetness, crunchiness, juiciness, ripeness, acidity, and grade. The results offer insights into the variability and the range of each attribute within the dataset.

THANK YOU!