DSBDA Assignment 2 Data Wrangling

Create/Choose a dataset of students and perform the following operations using python

1. Scan all variables for missing values and inconsistencies. If there are missing values and/or inconsistencies, use any of the suitable techniques to deal with them.

Missing Data ->

1. Missing completely at Random (MCAR)
   When the probability of missing data is unrelated to the precise value to the obtainded or the collection of the observed answers.
2. Missing at Random (MAR)
   When the probability of missing responses is decided by the collection of observed responses rather than the exact missing values expected to be reached.
3. Missing not at Random (MNAR)
   Other than the above-mentioned categories, MNART is the missing data. The MNAR data cases are a pain to deal with. Modelling the missing data is the only way to get a fair approximation fo the parameteres in this situaiton.

Data Frame:

	First Score	Second Score	Third Score
0	100.0	30.0	NaN
1	90.0	45.0	40.0
2	NaN	54.0	80.0
3	95.0	NaN	90.0

Checking for NaN Values:

```
data_frame.isnull().sum()
```

```
First Score     1
Second Score    1
Third Score     1
dtype: int64
```

fillna() method:

```
data_frame.fillna(0)
```

	First Score	Second Score	Third Score
0	100.0	30.0	0.0
1	90.0	45.0	40.0
2	0.0	54.0	80.0
3	95.0	0.0	90.0

```
data_frame.fillna(method='pad')
```

	First Score	Second Score	Third Score
0	100.0	30.0	NaN
1	90.0	45.0	40.0
2	90.0	54.0	80.0
3	95.0	54.0	90.0

```
data_frame.interpolate(method='linear', limit_direction='forward')
```

	First Score	Second Score	Third Score
0	100.0	30.0	NaN
1	90.0	45.0	40.0
2	92.5	54.0	80.0
3	95.0	54.0	90.0

```
data_frame.fillna(method='bfill')
```

	First Score	Second Score	Third Score
0	100.0	30.0	40.0
1	90.0	45.0	40.0
2	95.0	54.0	80.0
3	95.0	NaN	90.0

different ways to fill missing values

2. Scan all numeric varialbes for outliers. If there are outliers, use any of the suitable techniques to deal with them.

country_name	Continent	region	local_name	capital	area	population
Austria	Europe	Western Europe	Österreich	Vienna	83,879 km²	8,917,000
Belgium	Europe	Western Europe	België / Belgique	Brussels	30,530 km²	11,544,000
France	Europe	Western Europe	France	Paris	549,087 km²	67,380,000
Germany	Europe	Western Europe	Deutschland	Berlin	357,580 km²	83,161,000
Liechtenstein	Europe	Western Europe	Liechtenstein	Vaduz	161 km²	38,137

Population of Liechtenstein is very less, so an outlier for the data set.

Detecting Outlier:-

Method 1:- Using percentile Method.

```
max_threshold = data_frame['area'].quantile(0.90)
min_threshold = data_frame['area'].quantile(0.05)
print(max_threshold)
print(min_threshold)
```

```
filter_condition = (data_frame['area'] < min_threshold)
```

```
data_frame[filter_condition]
```

Outlier:-

4	4	Liechtenstein	Europe	Western Europe	Liechtenstein	Vaduz	161	38,137

Method 2:- Using Z Score Test

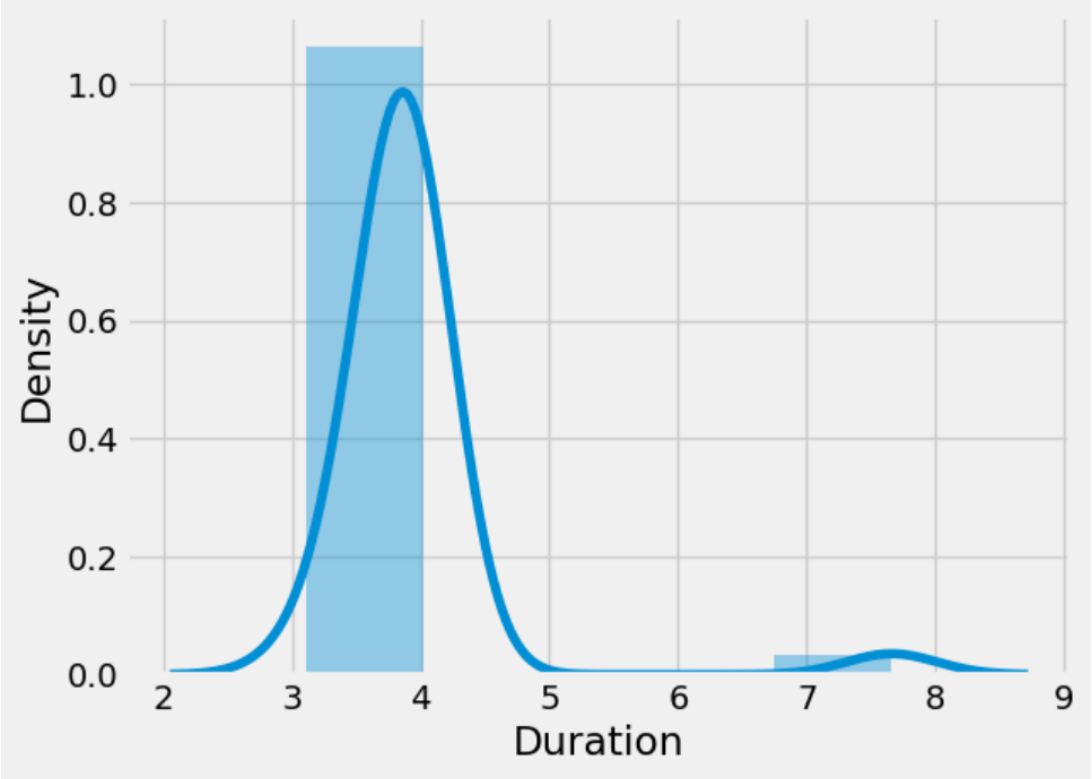
Dataset: calories.csv

```
df = pandas.read_csv('DataSets/calories.csv')
df
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.100
1	60	'2020/12/02'	117	145	479.000
2	60	'2020/12/03'	103	135	340.000
3	45	'2020/12/04'	109	175	282.400
4	45	'2020/12/05'	117	148	406.000
5	60	'2020/12/06'	102	127	300.000
6	60	'2020/12/07'	110	136	374.000
7	450	'2020/12/08'	104	134	253.300
8	30	'2020/12/09'	109	133	195.100
9	60	'2020/12/10'	98	124	269.000
10	60	'2020/12/11'	103	147	329.300

	Duration	Date	Pulse	Maxpulse	Calories	zscore
0	60	'2020/12/01'	110	130	409.100	-0.120
1	60	'2020/12/02'	117	145	479.000	-0.120
2	60	'2020/12/03'	103	135	340.000	-0.120
3	45	'2020/12/04'	109	175	282.400	-0.335
4	45	'2020/12/05'	117	148	406.000	-0.335
5	60	'2020/12/06'	102	127	300.000	-0.120
6	60	'2020/12/07'	110	136	374.000	-0.120
7	450	'2020/12/08'	104	134	253.300	5.448
8	30	'2020/12/09'	109	133	195.100	-0.549
9	60	'2020/12/10'	98	124	269.000	-0.120
10	60	'2020/12/11'	103	147	329.300	-0.120
11	60	'2020/12/12'	100	120	250.700	-0.120

Row 7 is an outlier for the data set.



Outlier :- Visual Represented

```
df['zscore'] = (df['Duration'] - df['Duration'].mean()) / df['Duration'].std()
```

Outlier Detectd:

```
outlier_filter = (df['zscore'] > 3) | (df['zscore'] < -3)
df[outlier_filter]
```

	Duration	Date	Pulse	Maxpulse	Calories	zscore
7	450	'2020/12/08'	104	134	253.300	5.448

3. Apply data transformation on at least one of the following reasons: to change the scale for better nderstanding of the variable, to convert a non linear relation into a linear one, or to decrease the skewness and convert the distribution into a normal distribution.

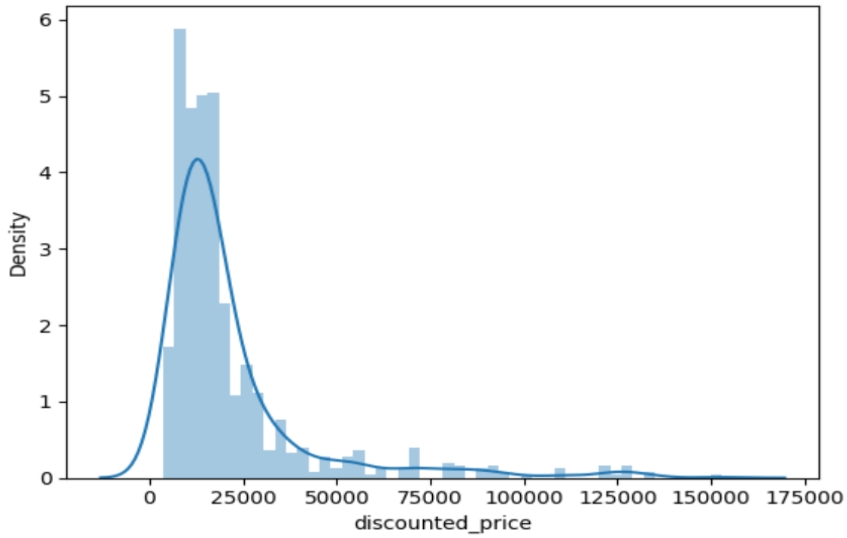
Skewness:- Skewness measure the distribution of the data It indicate weather the data is dirstributed symmetric or not. If the data is distributed symmetric means the data is normally distributed.

```
data_frame = pandas.read_csv('DataSets/flipkart_smartphones.csv')
```

```
df = data_frame[['brand', 'model', 'colour', 'original_price', 'discounted_price']]
```

```
df
```

	brand	model	colour	original_price	discounted_price
0	VIVO	VIVO T1 44W	Starry Sky	19990	14499
1	APPLE	APPLE IPHONE 11	White	48900	47199
2	VIVO	VIVO T1 44W	Midnight Galaxy	20990	15999
3	XIAOMI	POCO M4 5G	Power Black	15999	11999
4	XIAOMI	REDMI 10	Caribbean Green	14999	9299
...	...	...	...	...	...
831	REALME	REALME GT NEO 2	NEO Blue	38999	35999
832	REALME	REALME GT NEO 2	NEO Black	38999	35999
833	REALME	REALME GT NEO 2	NEO Black	34999	31999
834	REALME	REALME X50 PRO	Rust Red	17999	41999
835	XIAOMI	POCO M2 PRO	Two Shades of Black	17999	17999



Applying log transformation on skewed data

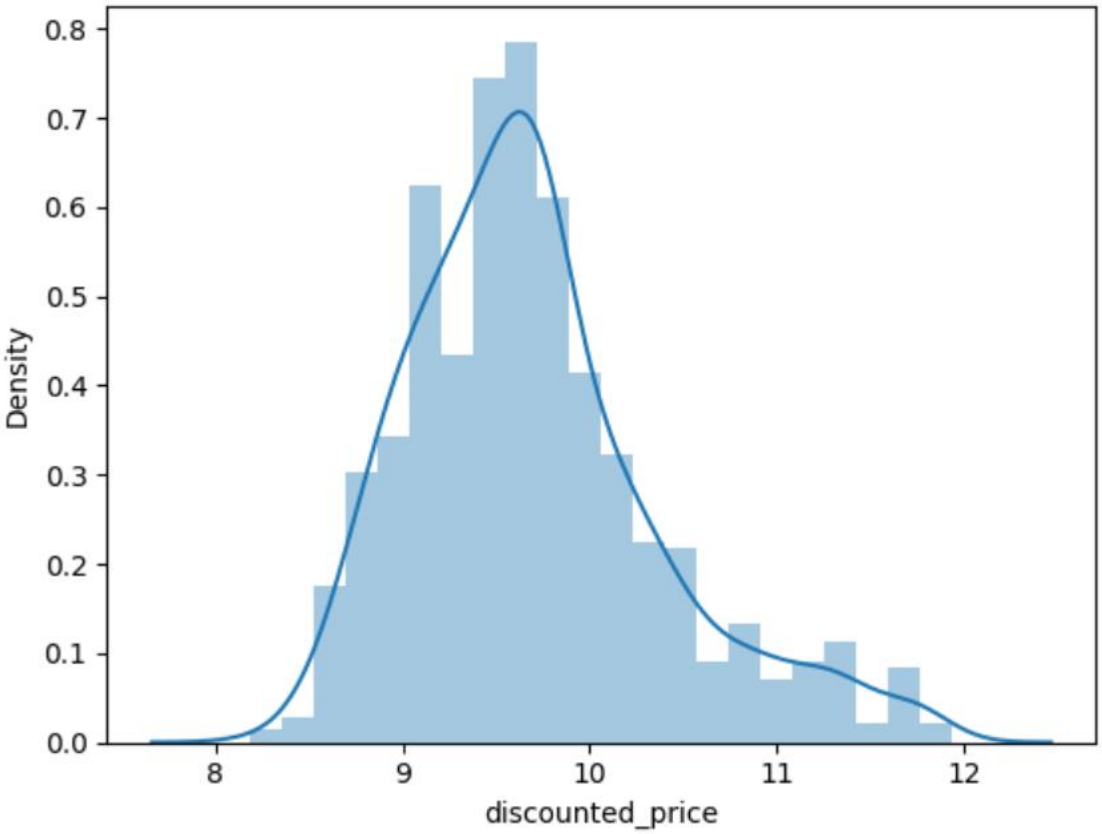
```
df['discounted_price'].skew()
```

```
3.0491976320782683
```

```
discounted_price = df['discounted_price']
discounted_price = numpy.log(discounted_price)
discounted_price.head(5)
discounted_price.skew()
```

```
0.879654441070323
```

After Applying Log Transformatin on skewed data



Skewness is tolerable.