AMAZON DATA ANALYSIS

DONE BY:

ABHISHEK V

ABHISHEK MISHRA

ANUSH P UPADHYA

INDEX

- Abstract
- Introduction
- The Dataset
- Data cleaning and Preprocessing
- Exploratory Data Analysis
- Hypothesis testing
- Results and conclusions

ABSTRACT

The data set mainly focuses on product listing on amazon.com: A dataset was selected from the internet with all the necessary requirements like necessary categorical variables, null/missing values or outliers to name a few. Henceforth, we clean the data by handling missing data or duplicate or irrelevant data or any outliers if present. The next step was exploratory data analysis (EDA) which includes a set of techniques to display data in such a way that interesting features will become apparent. This is followed by data visualization of the current data in the form of various graphs. We have represented line graph heat maps bar charts histograms using statics results obtained . Penultimately, we end it with normalization and standardization and discuss why it is needed. We conclude this with hypothesis testing and correlation.

INTRODUCTION

An Amazon product listing is the product page for each of the items you sell on Amazon. It is made up of the information you enter when you list your product including its title, images, description, and price. Shoppers on Amazon use product listing pages to make a purchase, i.e. the Add to Cart button is on all product listing pages. As a result, getting the product listing right will determine the success of selling your products on Amazon.

An Amazon product listing performs several functions, but the ones are:

- Enables your products to be found in Amazon searches
- Encourages shoppers to purchase your product more easily
- Helps them determine which product category is the most selling and which is the least and helps them make a business strategy accordingly.

THE DATASET

This dataset was obtained from kaggle (Product Listing From Amazon). This dataset originally includes around 30,000 records in it. It roughly had around 6-7% of NULL values. This dataset includes the different products bought in the month of October 2019 at (which was an offer period) different times of the day.

DESCRIPTION OF THE DATASET VARIABLES

- **Unique ID:** A32-digit unique ID for every product purchase made.
- **Crawl Timestamp:** Date and time of purchase
- **Category**: Category that a particular product belongs to
- **Product Title:** The name of the product
- **Description:** A short description about the product
- **Brand:** Brand name
- **Pack size/Quantity:** Quantity of the product
- **MRP:** The Maximum Retail Price of the product before discount
- **Price:** The discounted price of the product
- **Offers:** Discount given on a product
- **Combo Offers:** Shows the combo offers if available else shows a NaN value
- **Stock Availability:** Shows whether a product is on stock or not.

DATA CLEANING AND PREPROCESSING

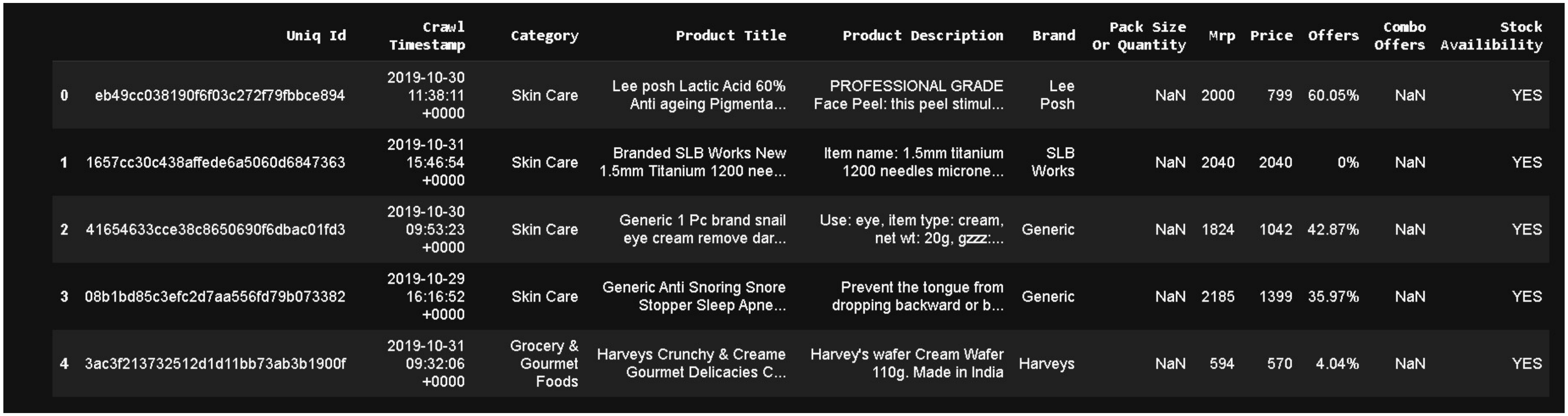
Data preprocessing involves the transformation of the raw dataset into an understandable format. Preprocessing data is a fundamental stage in Data Science and Machine Learning to improve data efficiency. The data preprocessing methods directly affect the outcomes of any analytic algorithm.

Steps Involved in Data Preprocessing:

1.Gathering the data
2.Import the dataset & Libraries
3.Dealing with Missing Values
4.Data Cleaning

Step 1: Gathering Data

Data is raw information, its the representation of both human and machine observation of the world. Dataset entirely depends on what type of problem we want to solve. Here we use a data set consisting of purchases made on amazon.com in the month of October,2019.

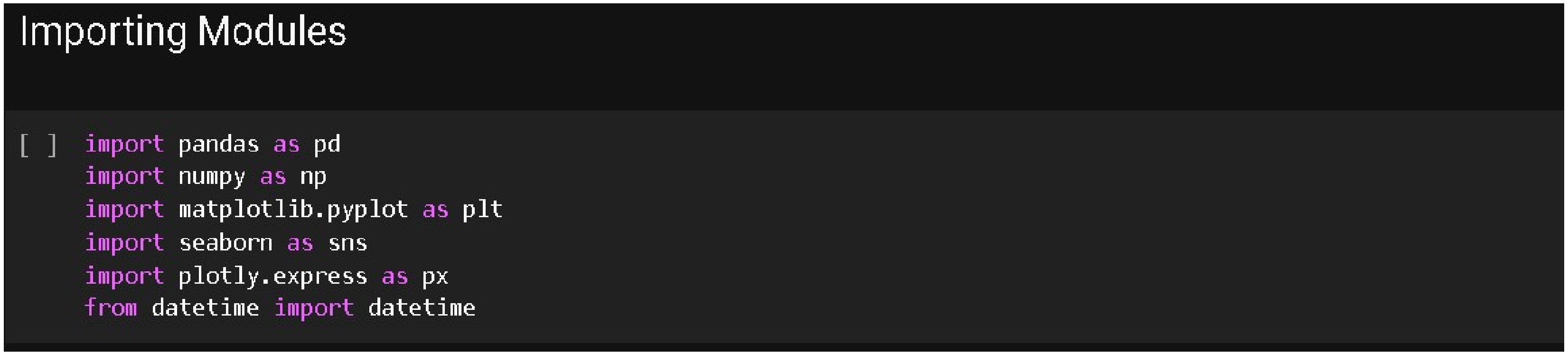


	Uniq Id	Crawl Timestamp	Category	Product Title	Product Description	Brand	Pack Size Or Quantity	Mrp	Price	Offers	Combo Offers	Stock Availibility
0	eb49cc038190f6f03c272f79fbbce894	2019-10-30 11:38:11 +0000	Skin Care	Lee posh Lactic Acid 60% Anti ageing Pigmenta...	PROFESSIONAL GRADE Face Peel: this peel stimul...	Lee Posh	NaN	2000	799	60.05%	NaN	YES
1	1657cc30c438affede6a5060d6847363	2019-10-31 15:46:54 +0000	Skin Care	Branded SLB Works New 1.5mm Titanium 1200 nee...	Item name: 1.5mm titanium 1200 needles microne...	SLB Works	NaN	2040	2040	0%	NaN	YES
2	41654633cce38c8650690f6dbac01fd3	2019-10-30 09:53:23 +0000	Skin Care	Generic 1 Pc brand snail eye cream remove dar...	Use: eye, item type: cream, net wt: 20g, gzzz:...	Generic	NaN	1824	1042	42.87%	NaN	YES
3	08b1bd85c3efc2d7aa556fd79b073382	2019-10-29 16:16:52 +0000	Skin Care	Generic Anti Snoring Snore Stopper Sleep Apne...	Prevent the tongue from dropping backward or b...	Generic	NaN	2185	1399	35.97%	NaN	YES
4	3ac3f213732512d1d11bb73ab3b1900f	2019-10-31 09:32:06 +0000	Grocery & Gourmet Foods	Harveys Crunchy & Creame Gourmet Delicacies C...	Harvey's wafer Cream Wafer 110g. Made in India	Harveys	NaN	594	570	4.04%	NaN	YES

Step 2: Importing the Dataset and Libraries

First step is usually importing the libraries that will be needed in the program. A library is essentially a collection of modules that can be called and used.

And can be import the libraries in python code with the help of ‘*import’* keyword.

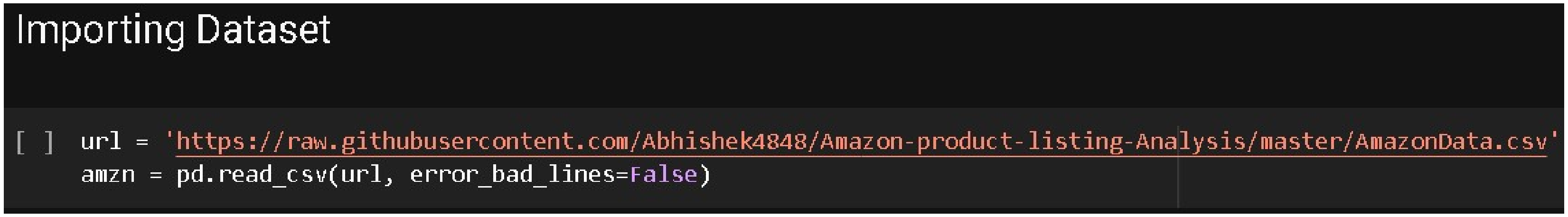


Importing Modules

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from datetime import datetime
```

Importing the dataset

Loading the data using Pandas library using *read_csv()* method.



Importing Dataset

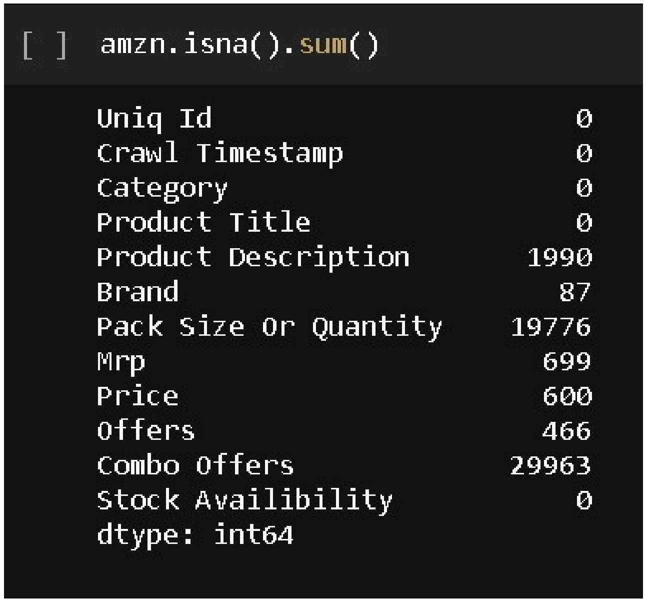
```
url = 'https://raw.githubusercontent.com/Abhishek4848/Amazon-product-listing-Analysis/master/AmazonData.csv'
amzn = pd.read_csv(url, error_bad_lines=False)
```

Step 3: Dealing with Missing Values

Sometimes we may find some data are missing in the dataset. if we found then we will remove those rows or we can calculate either mean, mode or median of the feature and replace it with missing values. This is an approximation which can add variance to the dataset.

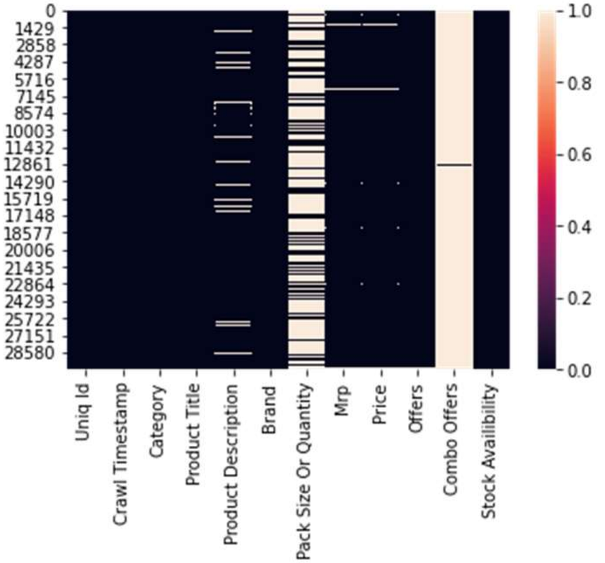
Check for null values:

we can check the null values in our dataset with pandas library as below.



```
[ ]  amzn.isna().sum()

     Uniq Id                     0
     Crawl Timestamp             0
     Category                    0
     Product Title               0
     Product Description      1990
     Brand                      87
     Pack Size Or Quantity   19776
     Mrp                       699
     Price                     600
     Offers                    466
     Combo Offers            29963
     Stock Availibility          0
     dtype: int64
```

Count of NaN/NULL values in dataset

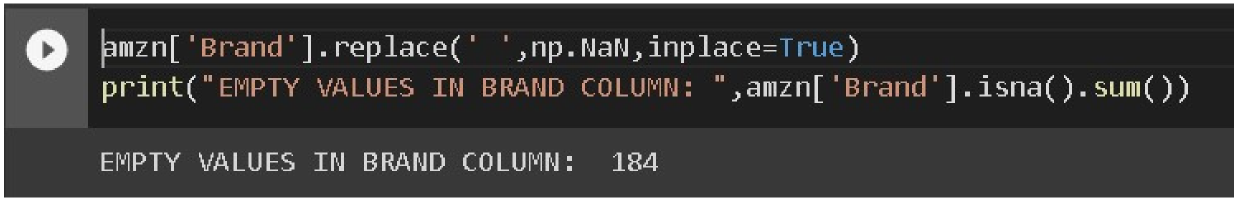


Heat Map visualizing NaN/NULL values in dataset

Step 4: Data Cleaning

Brand Column Fix

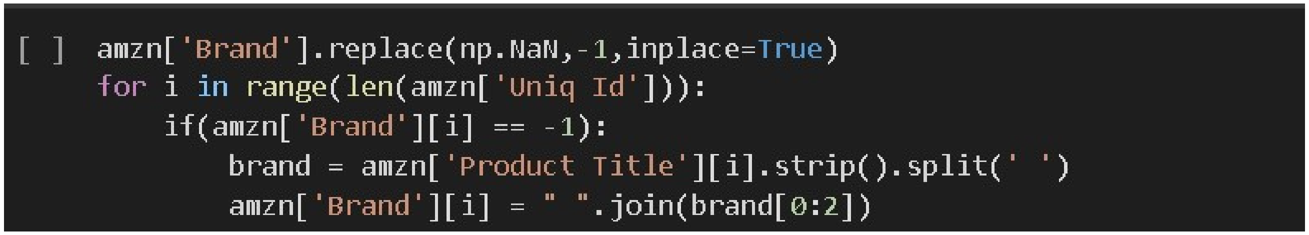
There are blank records in the Brand column that are as good as a NaN value.



```
amzn['Brand'].replace(' ',np.NaN,inplace=True)
print("EMPTY VALUES IN BRAND COLUMN: ",amzn['Brand'].isna().sum())
```

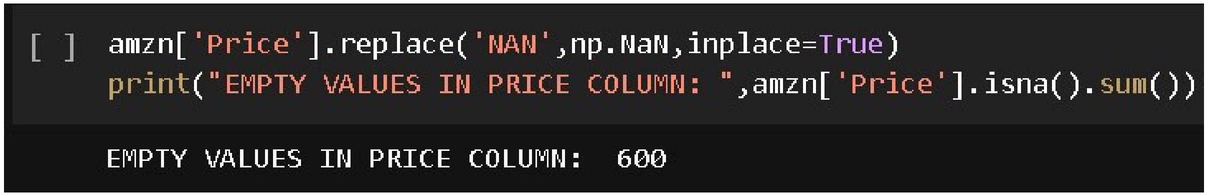
```
EMPTY VALUES IN BRAND COLUMN:  184
```

Since there is only 184 empty in the brand column , we can impute these values instead of removing it. We were able to get the brand name by performing simple string operations from the product description column.



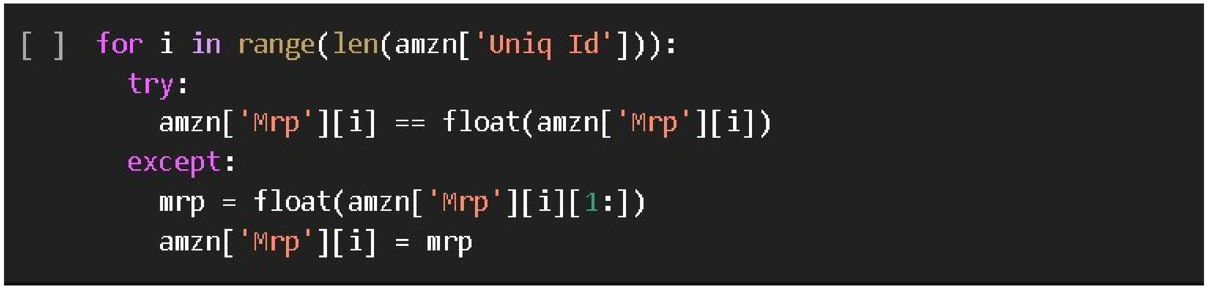
```
amzn['Brand'].replace(np.NaN,-1,inplace=True)
for i in range(len(amzn['Uniq Id'])):
    if(amzn['Brand'][i] == -1):
        brand = amzn['Product Title'][i].strip().split(' ')
        amzn['Brand'][i] = " ".join(brand[0:2])
```

Price and MRP column fix



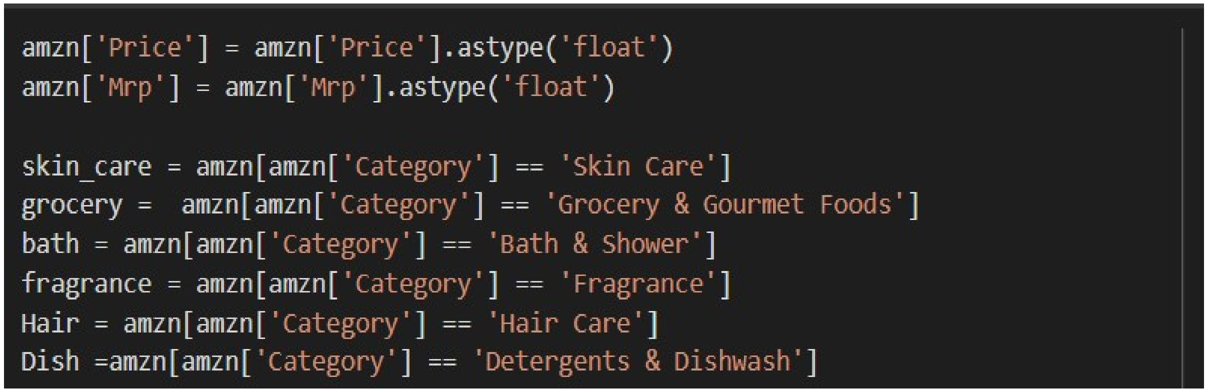
```
[ ]  amzn['Price'].replace('NAN',np.NaN,inplace=True)
     print("EMPTY VALUES IN PRICE COLUMN: ",amzn['Price'].isna().sum())

     EMPTY VALUES IN PRICE COLUMN:  600
```

→ We replace the string ‘nan’ with actual NumPy empty value.



```
[ ]  for i in range(len(amzn['Uniq Id'])):
       try:
         amzn['Mrp'][i] == float(amzn['Mrp'][i])
       except:
         mrp = float(amzn['Mrp'][i][1:])
         amzn['Mrp'][i] = mrp
```

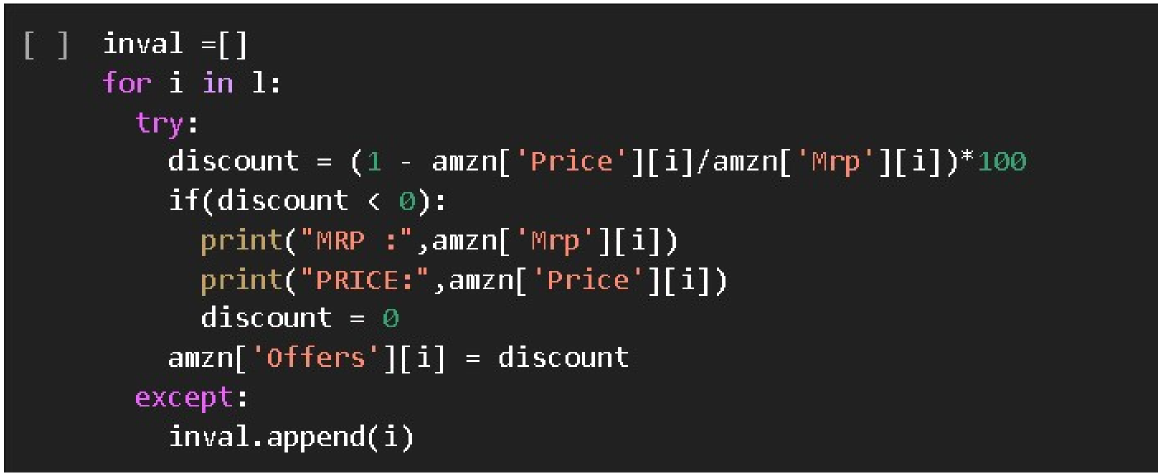
→ Then we convert the price and MRP columns to float value , since some of the data in these columns were in the form of string.



```
amzn['Price'] = amzn['Price'].astype('float')
amzn['Mrp'] = amzn['Mrp'].astype('float')

skin_care = amzn[amzn['Category'] == 'Skin Care']
grocery =  amzn[amzn['Category'] == 'Grocery & Gourmet Foods']
bath = amzn[amzn['Category'] == 'Bath & Shower']
fragrance = amzn[amzn['Category'] == 'Fragrance']
Hair = amzn[amzn['Category'] == 'Hair Care']
Dish =amzn[amzn['Category'] == 'Detergents & Dishwash']
```

→ Then we used mean and median imputation technique (based on different category) to fill the empty values in the MRP and Price column.

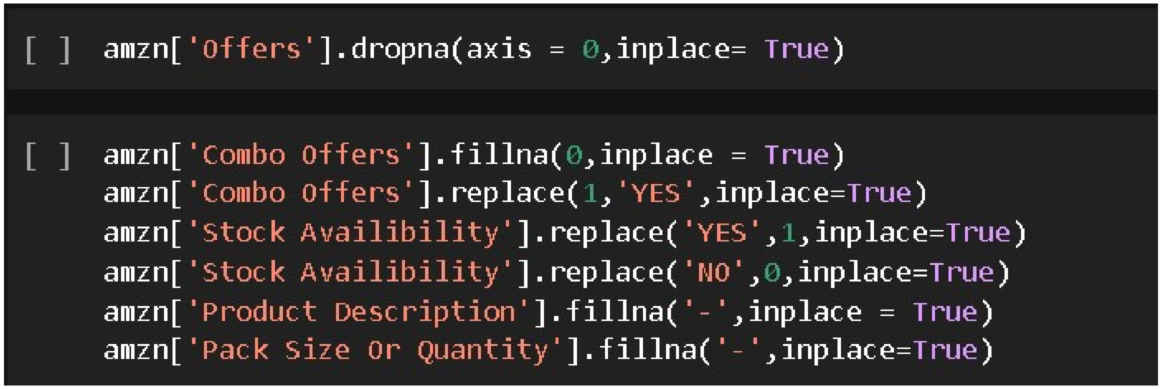
Offer Column Fix



```
[ ]  inval =[]
     for i in l:
       try:
         discount = (1 - amzn['Price'][i]/amzn['Mrp'][i])*100
         if(discount < 0):
           print("MRP :",amzn['Mrp'][i])
           print("PRICE:",amzn['Price'][i])
           discount = 0
         amzn['Offers'][i] = discount
       except:
         inval.append(i)
```

→ We replace the empty values in the offer columns by using the formula of offer price i.e.

Offer = (1 – price/Mrp)*100

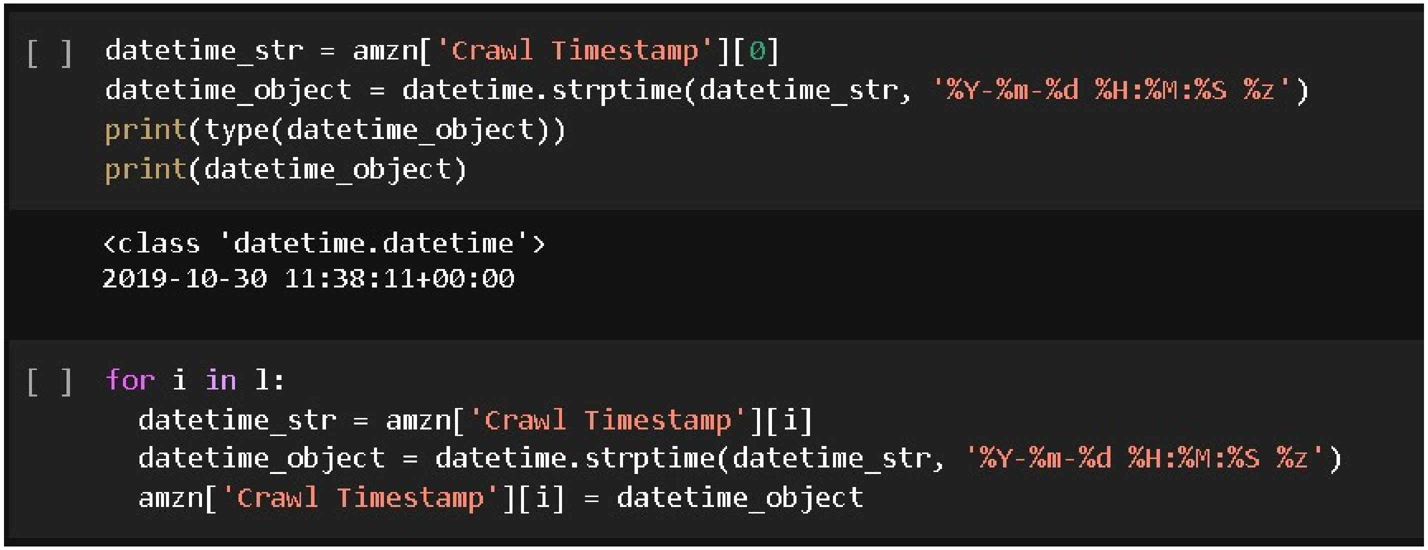


```
[ ]  amzn['Offers'].dropna(axis = 0,inplace= True)
```

```
[ ]  amzn['Combo Offers'].fillna(0,inplace = True)
     amzn['Combo Offers'].replace(1,'YES',inplace=True)
     amzn['Stock Availibility'].replace('YES',1,inplace=True)
     amzn['Stock Availibility'].replace('NO',0,inplace=True)
     amzn['Product Description'].fillna('-',inplace = True)
     amzn['Pack Size Or Quantity'].fillna('-',inplace=True)
```

→ We then replaces the category variable 'YES/NO' with binary 1 and 0 in the stock availability column.

Date Column fix



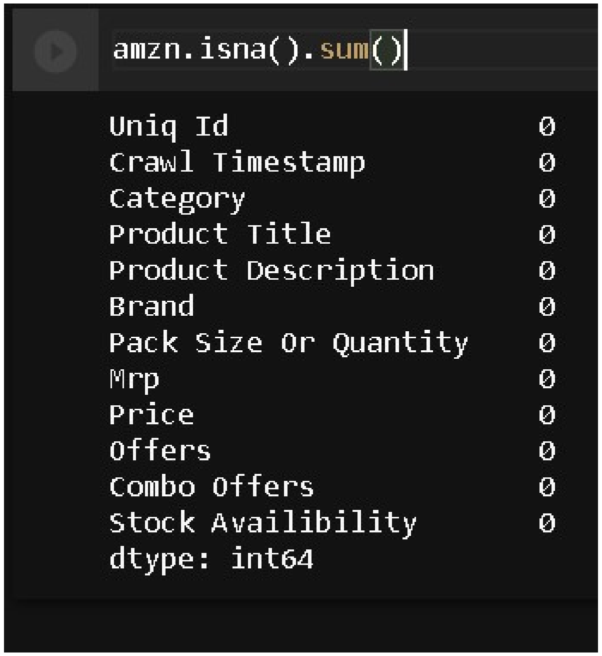
```python
datetime_str = amzn['Crawl Timestamp'][0]
datetime_object = datetime.strptime(datetime_str, '%Y-%m-%d %H:%M:%S %z')
print(type(datetime_object))
print(datetime_object)
```

```
<class 'datetime.datetime'>
2019-10-30 11:38:11+00:00
```

```python
for i in l:
  datetime_str = amzn['Crawl Timestamp'][i]
  datetime_object = datetime.strptime(datetime_str, '%Y-%m-%d %H:%M:%S %z')
  amzn['Crawl Timestamp'][i] = datetime_object
```

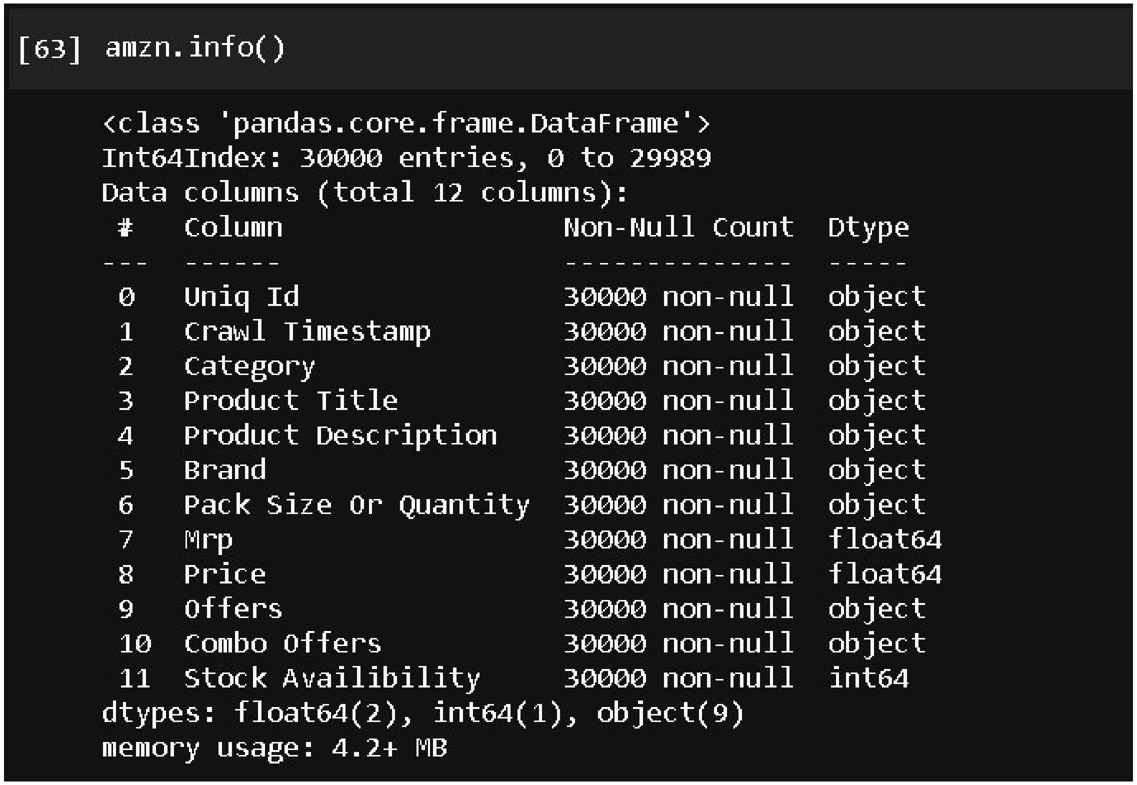
→We changed the date column to a timestamp object since it was in the form of a string. So that we were able to get days, years , time etc. easily this way.

Data After Cleaning and preprocessing



```
amzn.isna().sum()
```

```
Uniq Id                  0
Crawl Timestamp          0
Category                 0
Product Title            0
Product Description      0
Brand                    0
Pack Size Or Quantity    0
Mrp                      0
Price                    0
Offers                   0
Combo Offers             0
Stock Availibility       0
dtype: int64
```

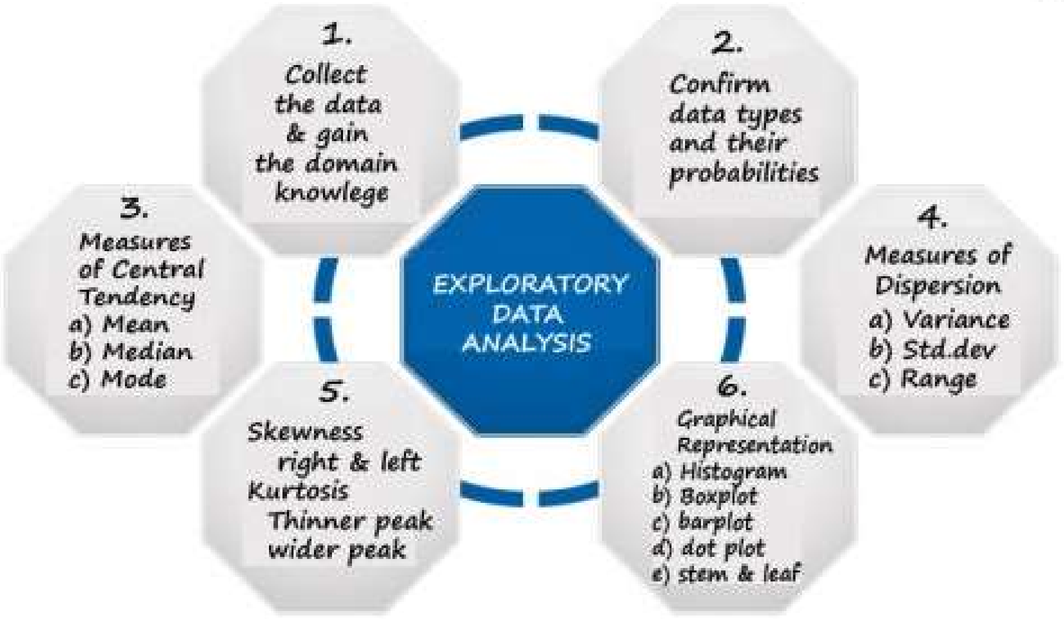


```
[63] amzn.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 30000 entries, 0 to 29989
Data columns (total 12 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   Uniq Id                30000 non-null  object 
 1   Crawl Timestamp        30000 non-null  object 
 2   Category               30000 non-null  object 
 3   Product Title          30000 non-null  object 
 4   Product Description    30000 non-null  object 
 5   Brand                  30000 non-null  object 
 6   Pack Size Or Quantity  30000 non-null  object 
 7   Mrp                    30000 non-null  float64
 8   Price                  30000 non-null  float64
 9   Offers                 30000 non-null  object 
 10  Combo Offers           30000 non-null  object 
 11  Stock Availibility     30000 non-null  int64  
dtypes: float64(2), int64(1), object(9)
memory usage: 4.2+ MB
```

EXPLORATORY DATA ANALYSIS (EDA)

Exploratory data analysis is the process of performing investigations on data so as to discover patterns, spot anomalies and to test for hypothesis with the help of summary statistics and graphical visualizations.



EXPLORATORY DATA ANALYSIS

1. Collect the data & gain the domain knowlege

2. Confirm data types and their probabilities

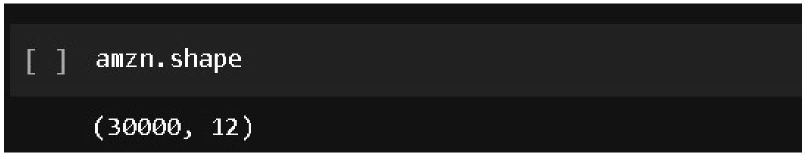
3. Measures of Central Tendency
a) Mean
b) Median
c) Mode

4. Measures of Dispersion
a) Variance
b) Std.dev
c) Range

5. Skewness
right & left
Kurtosis
Thinner peak
wider peak

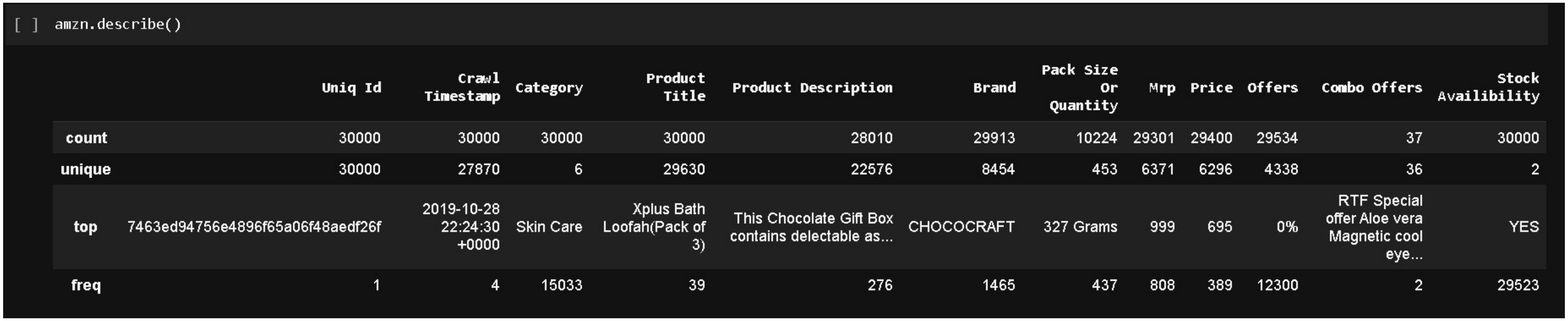
6. Graphical Representation
a) Histogram
b) Boxplot
c) barplot
d) dot plot
e) stem & leaf

Getting the No. of rows and columns in the data frame



```
[ ]  amzn.shape

     (30000, 12)
```

The describe() function in pandas is very handy in getting various summary statistics. This function returns the count, mean, standard deviation, minimum and maximum values and the quantiles of the data.

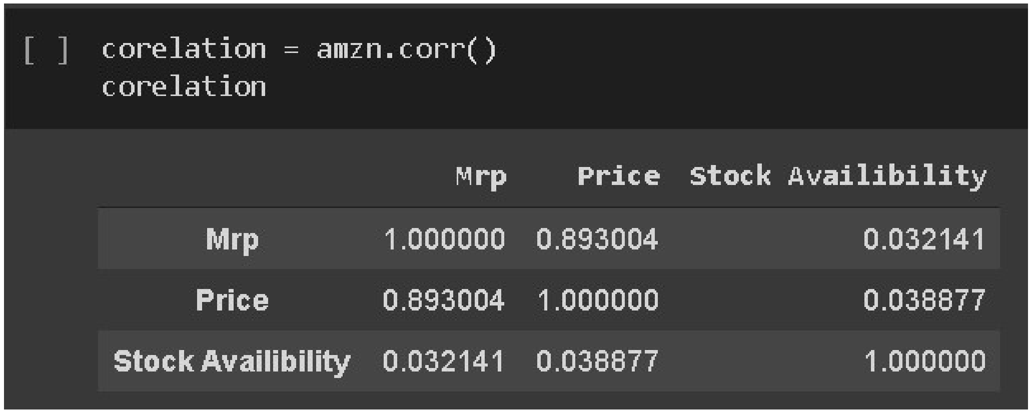


```
[ ]  amzn.describe()
```

	Uniq Id	Crawl Timestamp	Category	Product Title	Product Description	Brand	Pack Size Or Quantity	Mrp	Price	Offers	Combo Offers	Stock Availibility
count	30000	30000	30000	30000	28010	29913	10224	29301	29400	29534	37	30000
unique	30000	27870	6	29630	22576	8454	453	6371	6296	4338	36	2
top	7463ed94756e4896f65a06f48aedf26f	2019-10-28 22:24:30 +0000	Skin Care	Xplus Bath Loofah(Pack of 3)	This Chocolate Gift Box contains delectable as...	CHOCOCRAFT	327 Grams	999	695	0%	RTF Special offer Aloe vera Magnetic cool eye...	YES
freq	1	4	15033	39	276	1465	437	808	389	12300	2	29523

The Correlation Matrix

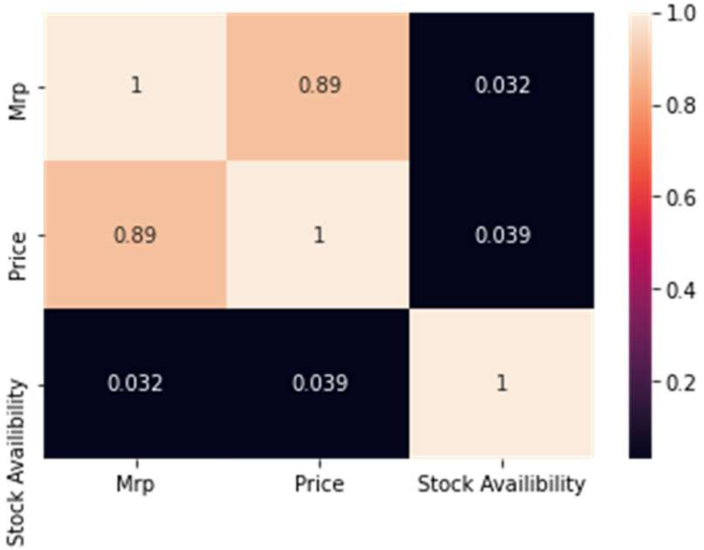
A correlation matrix is a table showing correlation coefficients between variables. Each cell in the table shows the correlation between two variables. A correlation matrix is used to summarize data, as an input into a more advanced analysis, and as a diagnostic for advanced analyses.



```
[ ]  corelation = amzn.corr()
     corelation
```

	Mrp	Price	Stock Availibility
Mrp	1.000000	0.893004	0.032141
Price	0.893004	1.000000	0.038877
Stock Availibility	0.032141	0.038877	1.000000

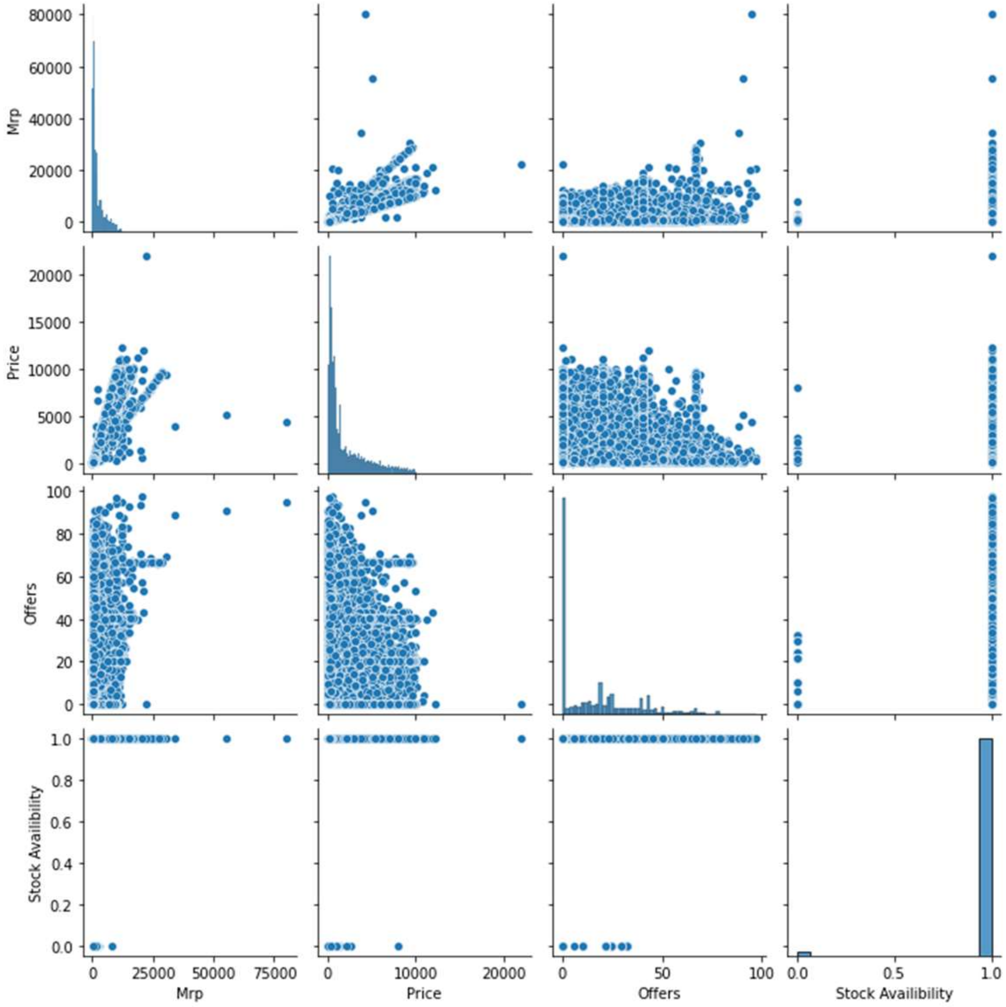
From the correlation matrix, we can infer that "MRP" has a strong correlation with "Price" whereas "MRP" and "Stock Availability" have a weak correlation.
This correlation is graphically represented using a heat map.



Heat Map visualizing the Correlation matrix

Pairplot

Pairplot from the Seaborn library helps plot multiple pairwise bivariate distributions in a dataset. This shows the relationship for (n, 2) combination of variable in a DataFrame as a matrix of plots and the diagonal plots are the univariate plots.



GRAPH VISUALIZATIONS

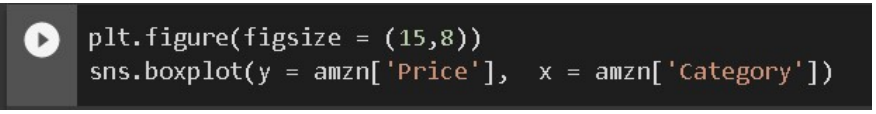
1. Boxplot

A box plot (or box-and-whisker plot) shows the distribution of quantitative data in a way that facilitates comparisons between variables. The box shows the quartiles of the dataset while the whiskers extend to show the rest of the distribution.

The box plot is a standardized way of displaying the distribution of data based on the five number summary:

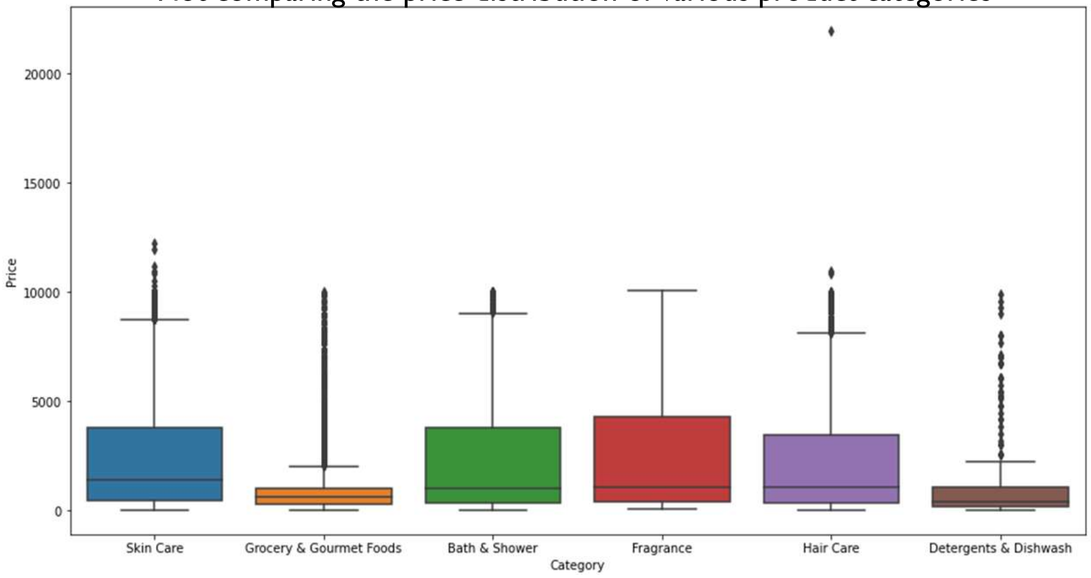
•Minimum
•First quartile
•Median
•Third quartile
•Maximum.

The boxplot from the Seaborn library helps visualize this distribution



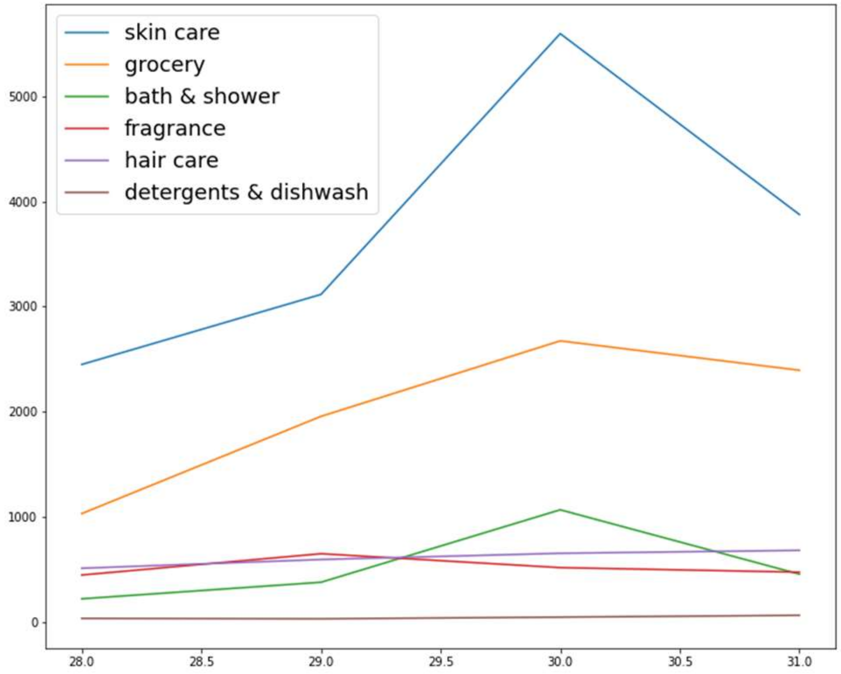
```
plt.figure(figsize = (15,8))
sns.boxplot(y = amzn['Price'],  x = amzn['Category'])
```

Plot comparing the price distribution of various product categories



2. Line Graph

The lineplot from seaborn library measures change over time by plotting individual data points connected by straight lines.

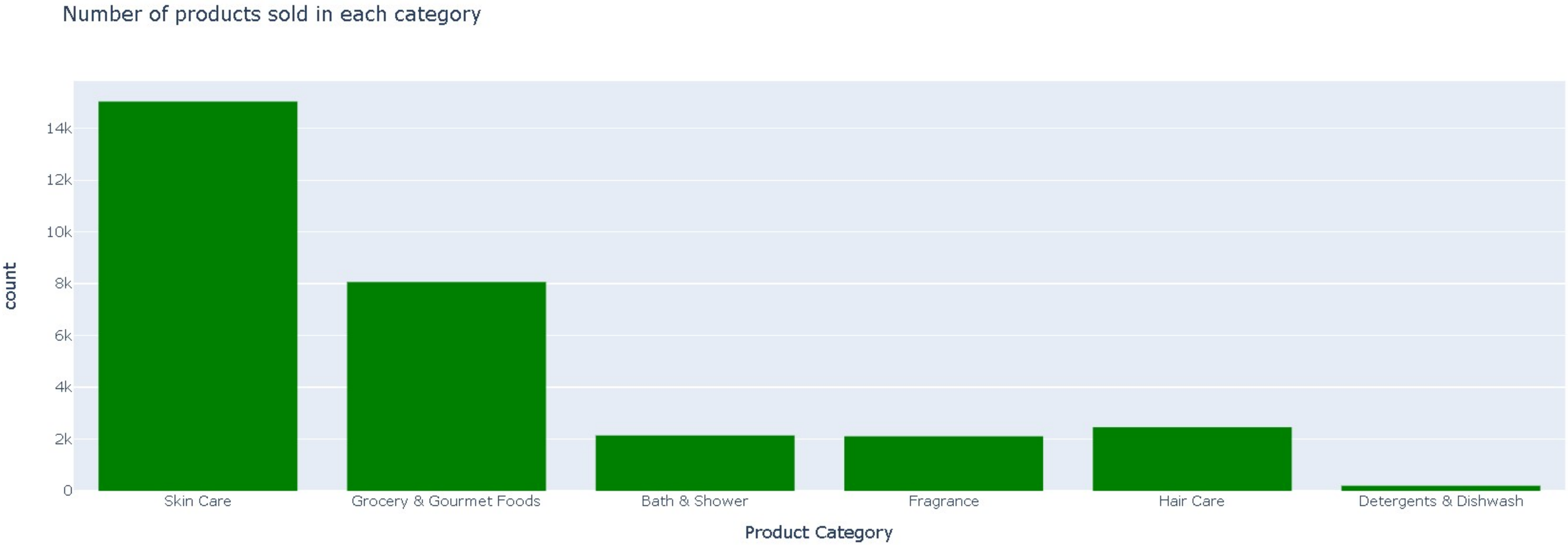


From the above line graph we can infer that Skin care products have a higher sales margin than other products listed. It is also evident that Skin Care products sales decreases in the last few days of the month.

The sales of of Detergents & Dishwashing products remain constant at the lower end throughout the month.

2. Histograms

Histograms provide a visual interpretation of numerical data by indicating the number of data points that lie within a range of values.

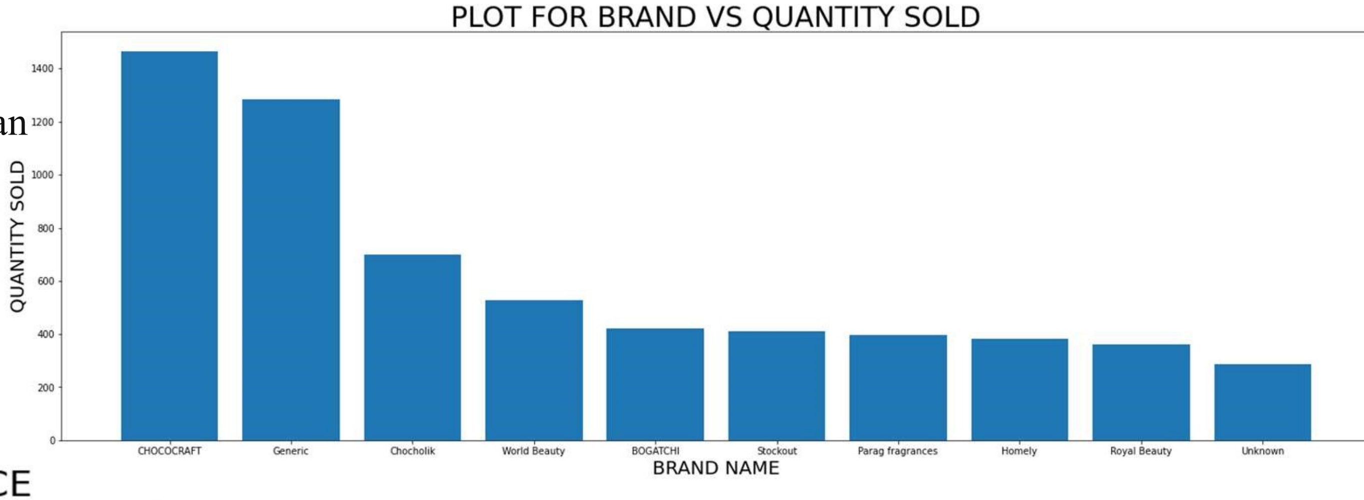


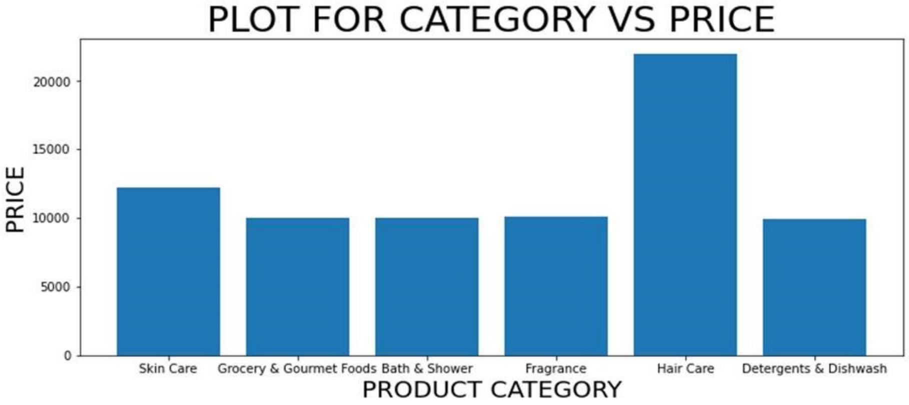
From the histogram representing the number of products sold in each category, one can infer that Skin Care products are the highest selling products in the month with over 15,000 units sold and Detergents & Dishwash are the least selling products with less than 200 units sold in the month.

3. Bar Graph

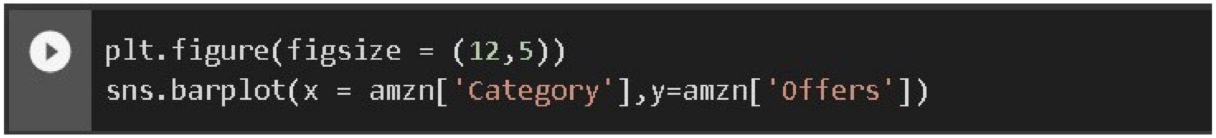
The bar plot from matplotlib.pyplot library helps plot data in rectangular bins that represent the total amount of observations in the data for that category.

From this bar graph representing the number of items sold by top brands, we can infer that “Chococraft” is the best selling brand in the month.





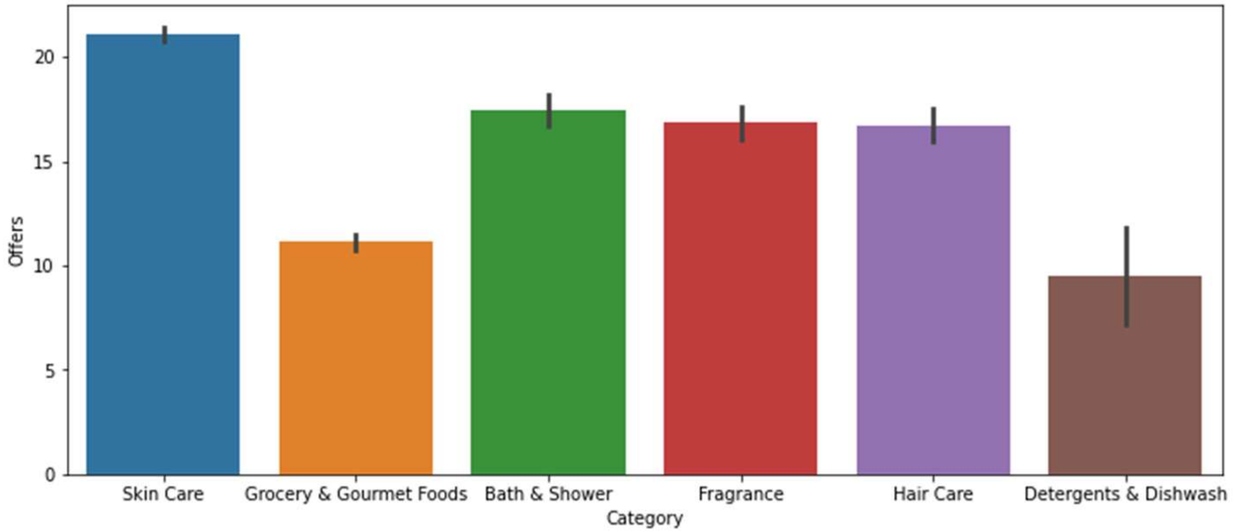
From this bar graph “Product category” vs “Price”, it is evident that hair care products are more expensive than the other products.



```
plt.figure(figsize = (12,5))
sns.barplot(x = amzn['Category'],y=amzn['Offers'])
```

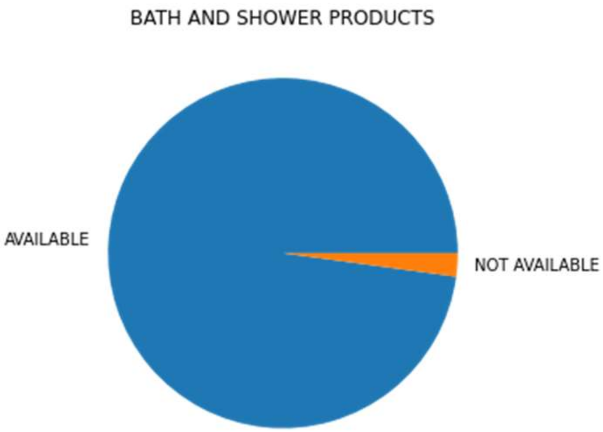
We can infer that skin care products has the highest offers which may also be a reason for increased sale of skin care products

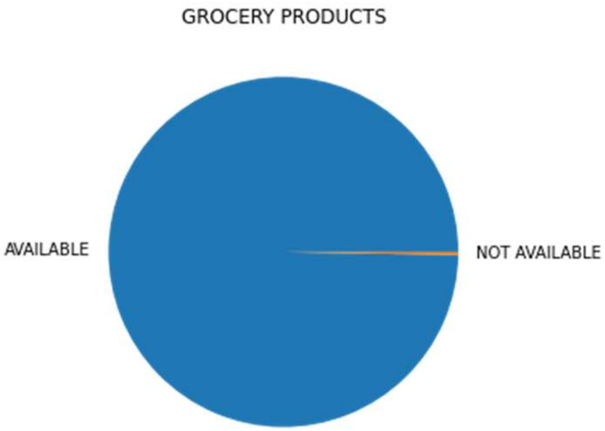
Bar Graph representing offers on various product categories

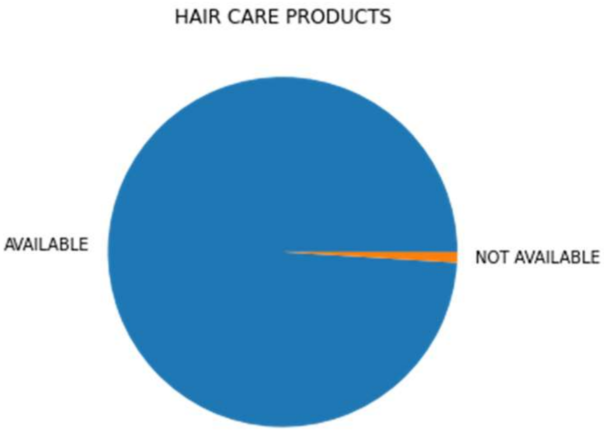


4. Pie Chart

A pie chart is a circular chart that shows how data sets relate to one another. The arc length of each section is proportional to the quantity it represents, usually resulting in a shape similar to a slice of pie







From the pie chart representing the stock availability of products under different categories, it is evident that majority of the products are on stock and very less number of products are out of stock. Products under the “Bath and Shower” category has slight higher percentage of unavailability.

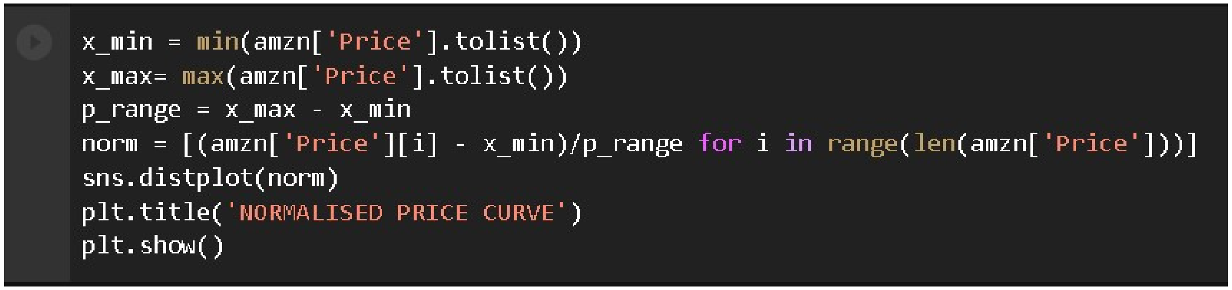
NORMALIZATION

In normalization, we convert the data features of different scales to a common scale which further makes it easy for the data to be processed for modeling.Thus, all the data features(variables) tend to have a similar impact on the modeling portion.

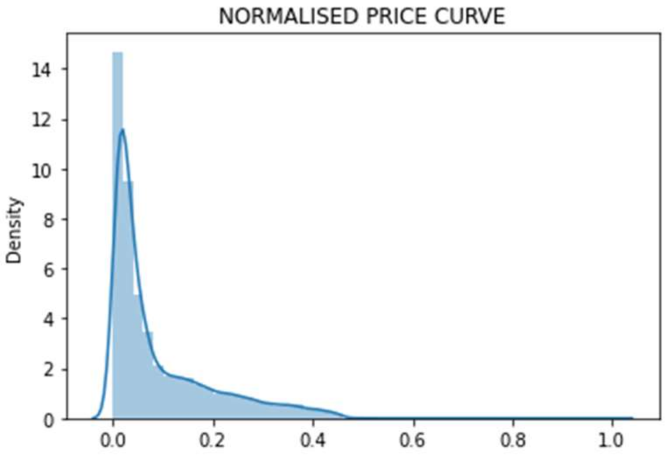
According to the below formula, we normalize each feature by subtracting the minimum data value from the data variable and then divide it by the range of the variable as shown–

$$x_{\text{norm}} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

The product price is normalized using the above formula



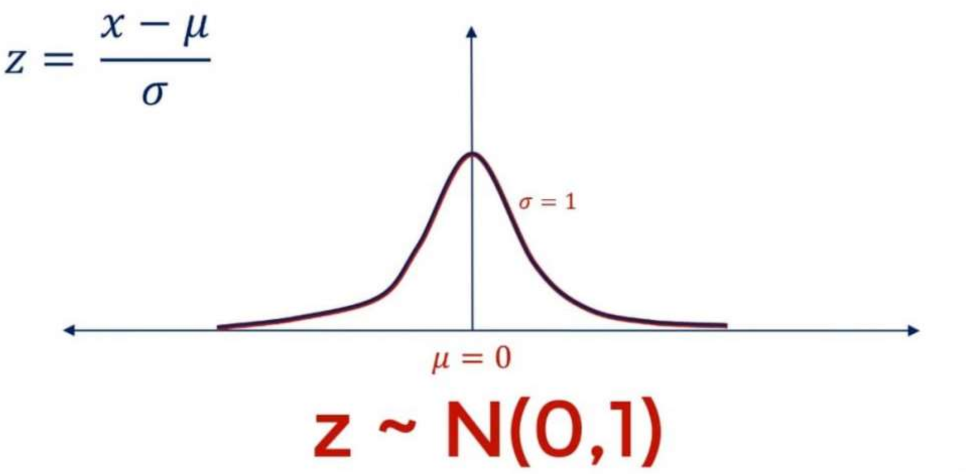
```
x_min = min(amzn['Price'].tolist())
x_max= max(amzn['Price'].tolist())
p_range = x_max - x_min
norm = [(amzn['Price'][i] - x_min)/p_range for i in range(len(amzn['Price']))]
sns.distplot(norm)
plt.title('NORMALISED PRICE CURVE')
plt.show()
```



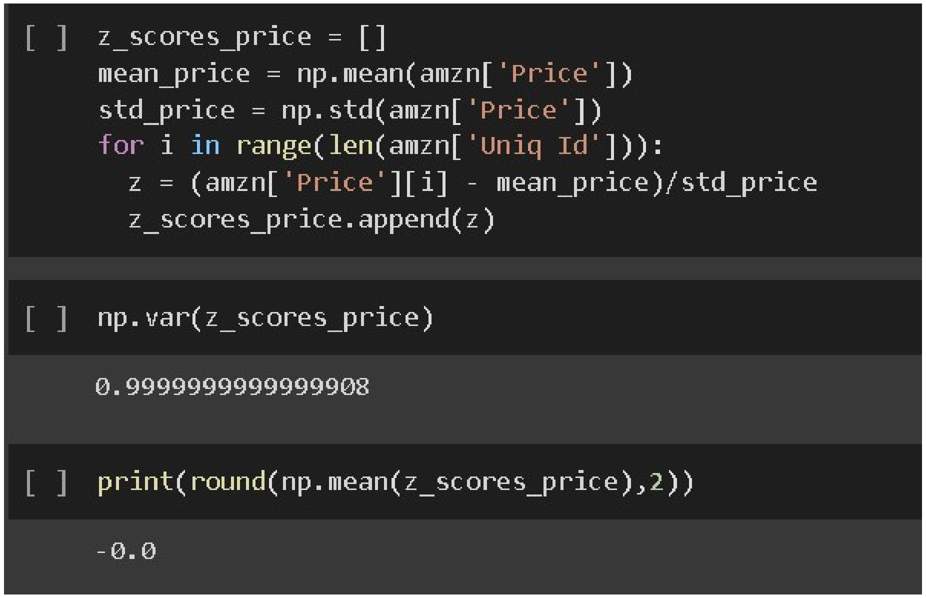
STANDARDIZATION

Data standardization is about making sure that data is internally consistent; that is, each data type has the same content and format. It transforms data to have a mean of 0 and standard deviation of 1.

Standardized data is essential for accurate data analysis; it's easier to draw clear conclusions about your current data when you have other data to measure it against.



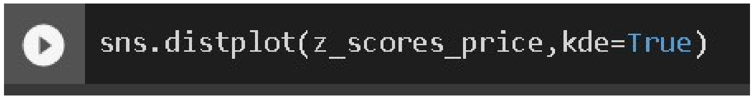
Standardizing the product prices:



```
[ ]  z_scores_price = []
     mean_price = np.mean(amzn['Price'])
     std_price = np.std(amzn['Price'])
     for i in range(len(amzn['Uniq Id'])):
       z = (amzn['Price'][i] - mean_price)/std_price
       z_scores_price.append(z)
```

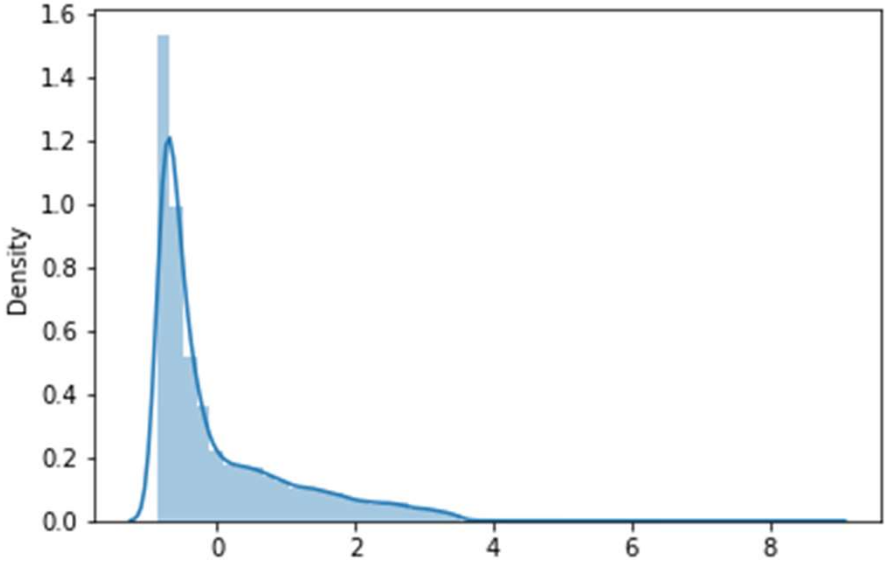
```
[ ]  np.var(z_scores_price)

     0.9999999999999908
```

```
[ ]  print(round(np.mean(z_scores_price),2))

     -0.0
```



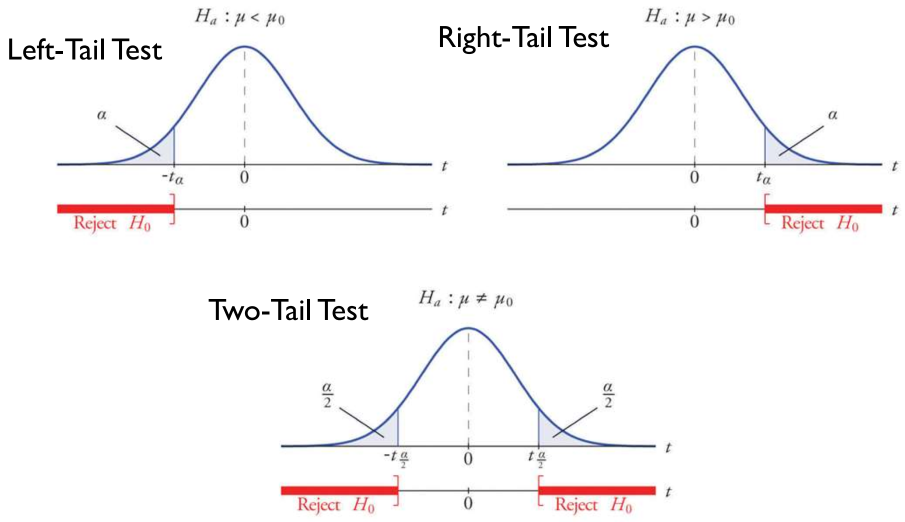
```
sns.distplot(z_scores_price,kde=True)
```

The kde parameter is set to True to enable the Kernel Density Plot along with the distplot



HYPOTHESIS TESTING

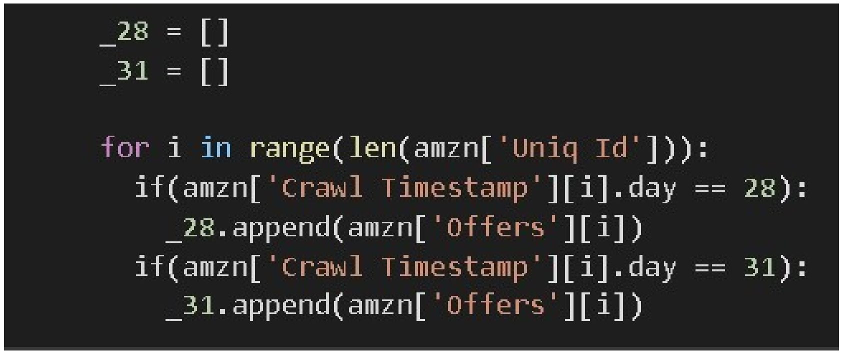
Hypothesis testing is a form of statistical inference that uses data from a sample to draw conclusions about a population parameter or a population probability distribution. First, a tentative assumption is made about the parameter or distribution. This assumption is called the null hypothesis and is denoted by H_0. An alternative hypothesis (denoted H_a), which is the opposite of what is stated in the null hypothesis, is then defined. The hypothesis-testing procedure involves using sample data to determine whether or not H_0 can be rejected. If H_0 is rejected, the statistical conclusion is that the alternative hypothesis H_a is true.



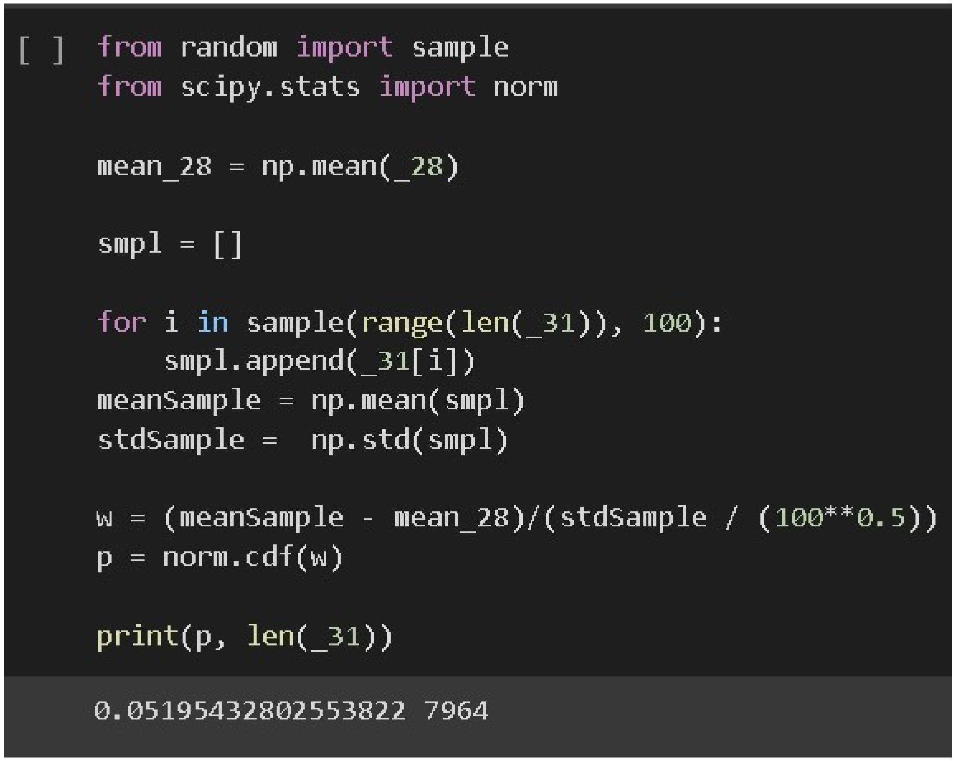
Hypothesis: To check whether the offers increases with time

We assume the significance level to be 5% = 0.05
Average price on 28th: x
Average price on 31st: mu0
Null hypothesis H0: mu0 <= x
Alternate hypothesis: mu0 > x

Formula: $z = (x — \mu) / (\sigma / \sqrt{n})$



```
_28 = []
_31 = []

for i in range(len(amzn['Uniq Id'])):
  if(amzn['Crawl Timestamp'][i].day == 28):
    _28.append(amzn['Offers'][i])
  if(amzn['Crawl Timestamp'][i].day == 31):
    _31.append(amzn['Offers'][i])
```



```
[ ]  from random import sample
     from scipy.stats import norm

     mean_28 = np.mean(_28)

     smpl = []

     for i in sample(range(len(_31)), 100):
         smpl.append(_31[i])
     meanSample = np.mean(smpl)
     stdSample =  np.std(smpl)

     w = (meanSample - mean_28)/(stdSample / (100**0.5))
     p = norm.cdf(w)

     print(p, len(_31))
```

```
0.05195432802553822 7964
```

We obtain the value of P as 0.05195 which is greater than the significance level(0.05). Hence we accept the null hypothesis. We accept the Null Hypothesis i.e. we cannot say that the offers have increased from 28th to 31st.

RESULTS AND CONCLUSIONS

→ Since this data was collected during one of the offer and festival seasons , it was observed that Skin care products were the highest selling products with high stock availability and lot of offers.

→We were also able to conclude based on the analysis that the sale of a product depends mainly on two factors that are :

- ❑Stock availability of the products
- ❑The offers available on the product.

Thank You

View our project in Google Colab:
https://colab.research.google.com/drive/107wwi3RH1nrsN8JCz9fJus7ENzNqCyiM?usp=sharing
Link to Github repository: https://github.com/Abhishek4848/Amazon-product-listing-Analysis