Data Collection and Preprocessing Phase

Date	10 July 2024
Team ID	739921
Project Title	Smartwatch Price Prediction
Maximum Marks	6 Marks

Data Exploration and Preprocessing Report

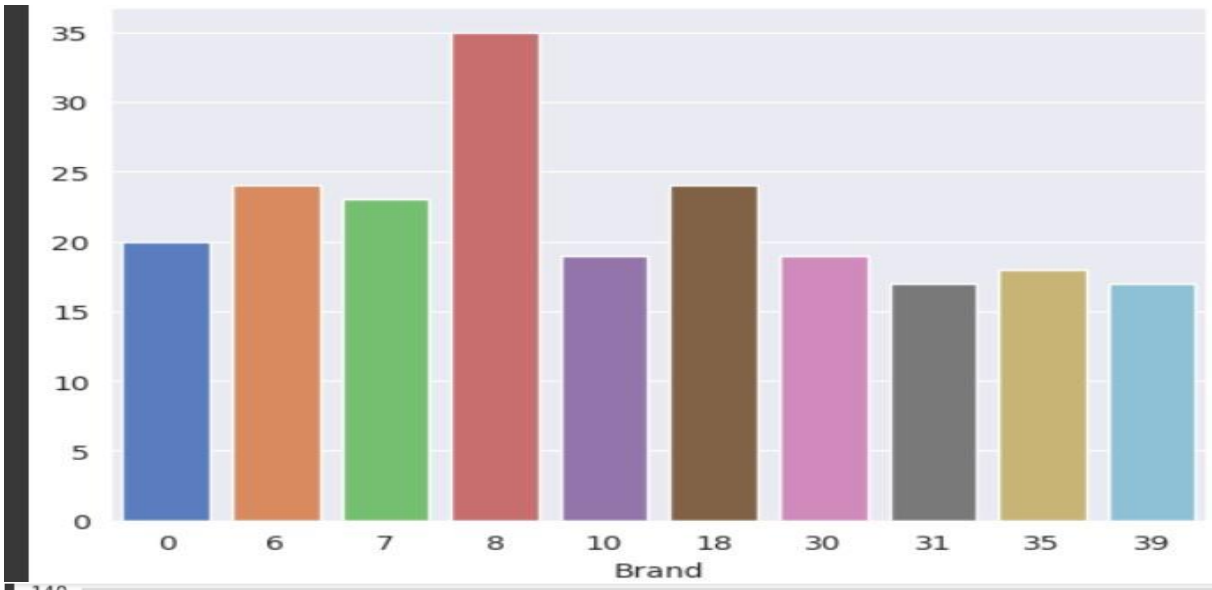
Dataset variables will be statistically analyzed to identify patterns and outliers, with Python employed for preprocessing tasks like normalization and feature engineering. Data cleaning

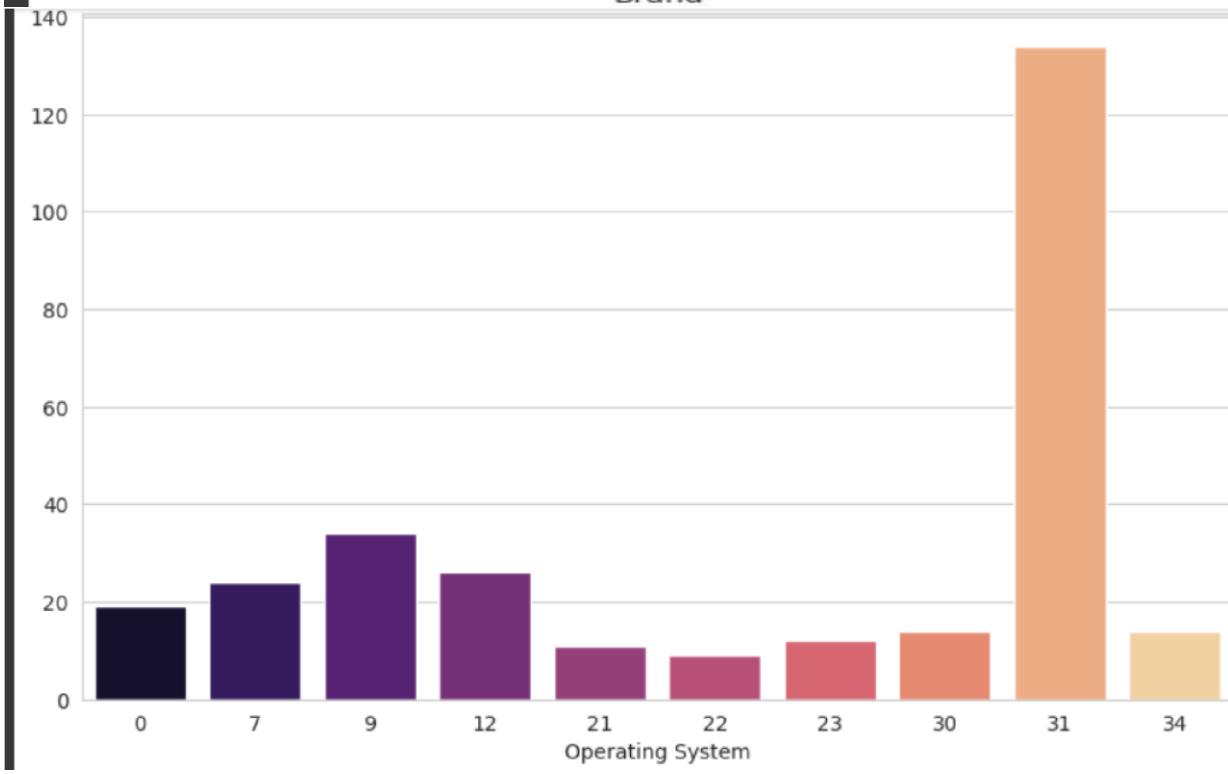
Section	Description
Data Overview	Dimension: 379 rows × 13 columns Descriptive statistics:

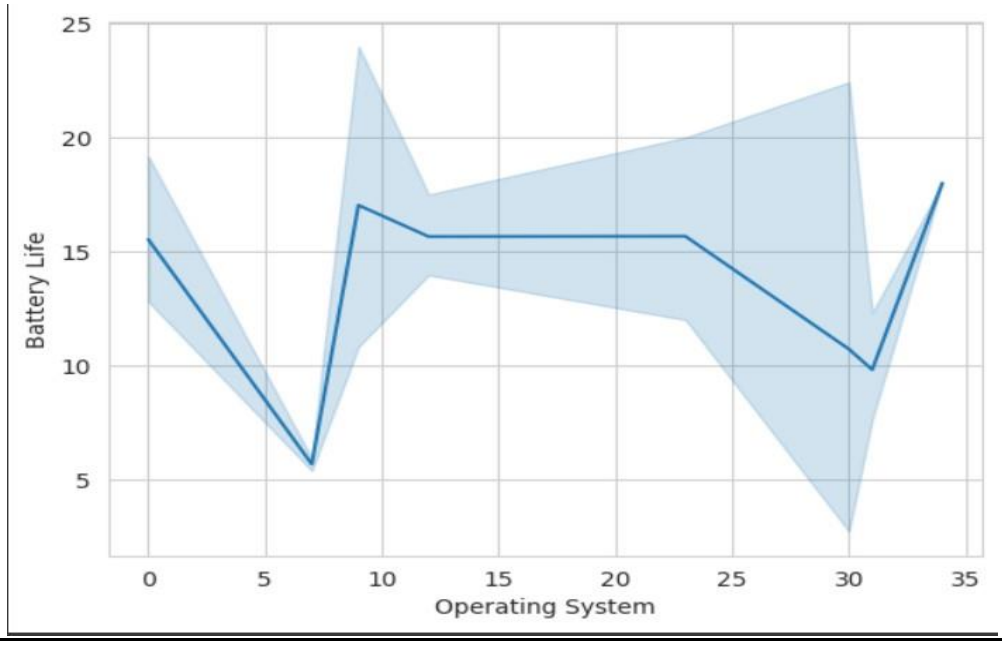
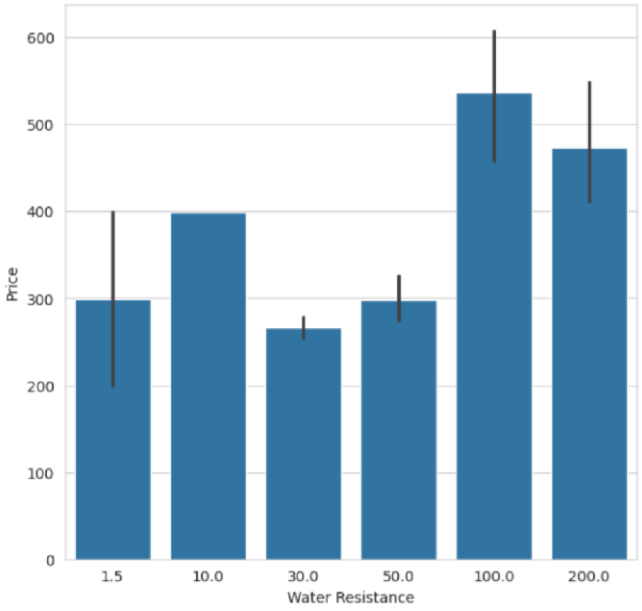
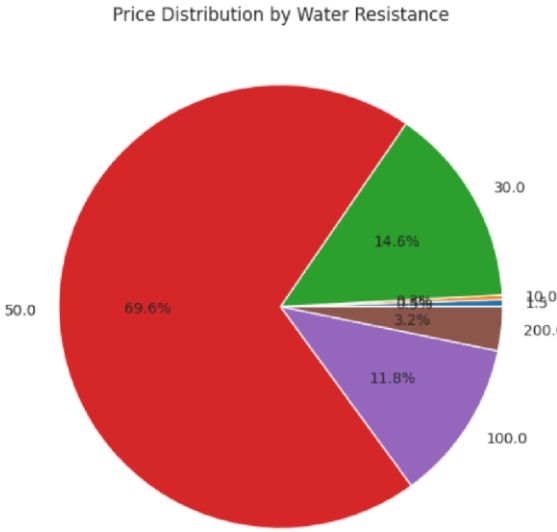
	Brand	Model	Operating System	Connectivity	Display Type	Display Size	Resolution	Water Resistance	Battery Life	Heart Rate Monitor	GPS	NFC
count	379.000000	379.000000	379.000000	379.000000	379.000000	379.000000	379.000000	379.000000	379.000000	379.0	379.000000	379.00000
mean	18.168865	68.606860	20.778364	1.203166	6.941953	1.368074	22.139842	52.804749	12.208443	0.0	0.920844	0.83905
std	13.040757	38.933753	11.407946	0.532927	8.978918	0.219087	9.080415	26.939235	12.326042	0.0	0.270338	0.36797
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.900000	0.000000	1.500000	1.000000	0.0	0.000000	0.00000
25%	7.000000	33.500000	9.000000	1.000000	0.000000	1.200000	17.500000	50.000000	3.000000	0.0	1.000000	1.00000
50%	16.000000	71.000000	27.000000	1.000000	0.000000	1.400000	23.000000	50.000000	11.000000	0.0	1.000000	1.00000
75%	31.000000	102.000000	31.000000	1.000000	14.000000	1.400000	32.000000	50.000000	15.000000	0.0	1.000000	1.00000
max	41.000000	136.000000	34.000000	4.000000	26.000000	4.000000	35.000000	200.000000	72.000000	0.0	1.000000	1.00000

will address missing values and outliers, ensuring quality for subsequent analysis and modeling, and forming a strong foundation for insights and predictions.

Univariate Analysis





Bivariate Analysis	
Multivariate Analysis	 
Outliers and Anomalies	-

Data Preprocessing Code Screenshots

Loading Data

```
[77] df =pd.read_csv("/content/Smart watch prices.csv")
```

```
df
```

	Brand	Model	Operating System	Connectivity	Display Type	Display Size (inches)	Resolution	Water Resistance (meters)	Battery Life (days)	Heart Rate Monitor	GPS	NFC	Price (USD)
0	Apple	Watch Series 7	watchOS	Bluetooth, Wi-Fi, Cellular	Retina	1.90	396 x 484	50	18	Yes	Yes	Yes	$399
1	Samsung	Galaxy Watch 4	Wear OS	Bluetooth, Wi-Fi, Cellular	AMOLED	1.40	450 x 450	50	40	Yes	Yes	Yes	$249
2	Garmin	Venu 2	Garmin OS	Bluetooth, Wi-Fi	AMOLED	1.30	416 x 416	50	11	Yes	Yes	No	$399
3	Fitbit	Versa 3	Fitbit OS	Bluetooth, Wi-Fi	AMOLED	1.58	336 x 336	50	6	Yes	Yes	Yes	$229
4	Fossil	Gen 6	Wear OS	Bluetooth, Wi-Fi	AMOLED	1.28	416 x 416	30	24	Yes	Yes	Yes	$299
...	...	...	...	...	...	...	...	...	...	...	...	...	...
374	Withings	ScanWatch	Withings OS	Bluetooth, Wi-Fi	PMOLED	1.38	348 x 442	50	30	Yes	No	Yes	$279
375	Zepp	Z	Zepp OS	Bluetooth, Wi-Fi, Cellular	AMOLED	1.39	454 x 454	50	15	Yes	Yes	Yes	$349
376	Honor	Watch GS Pro	Lite OS	Bluetooth, Wi-Fi	AMOLED	1.39	454 x 454	50	25	Yes	Yes	Yes	$249
377	Oppo	Watch Free	ColorOS	Bluetooth, Wi-Fi	AMOLED	1.64	326 x 326	50	14	Yes	No	Yes	$159

Handling Missing Data

```
df.isna().sum()
```

```
Brand                      1
Model                      1
Operating System           3
Connectivity               1
Display Type               2
Display Size (inches)      3
Resolution                 4
Water Resistance (meters)  1
Battery Life (days)        1
Heart Rate Monitor         1
GPS                        1
NFC                        1
Price (USD)                1
dtype: int64
```

```
[80] object_columns = df.select_dtypes(include=["object"]).columns
     for col in object_columns:
       mode_value = df[col].mode()[0]
       df[col] = df[col].fillna(mode_value)
```

```
[81] float_columns = df.select_dtypes(include=["float64"]).columns
     for col in float_columns:
       mean_value = df[col].mean()
       df[col] = df[col].fillna(mean_value)
```

Data Transformation

```
df = df.rename(columns={
    'Display Size (inches)':'Display Size',
    'Water Resistance (meters)':'Water Resistance',
    'Battery Life (days)':'Battery Life',
    'Price (USD)':'Price'
})
```

```
df ['Water Resistance'].unique()
```

```
array(['50', '30', '100', '1.5', 'Not specified', '200', '10'],
      dtype=object)
```

```
df ['Water Resistance'].describe()
```

```
count     379
unique      7
top        50
freq      276
Name: Water Resistance, dtype: object
```

```
df['Water Resistance']= df['Water Resistance'].replace({'Not specified': '50'})
```

```
df['Display Size'].unique()
```

```
array([1.9       , 1.4       , 1.3       , 1.58      , 1.28      ,
       1.43      , 1.75      , 1.39      , 1.36316489, 1.65      ,
       1.2       , 1.57      , 1.        , 1.78      , 1.91      ,
       1.38      , 1.06      , 1.35      , 1.34      , 0.9       ,
       1.04      , 1.64      , 1.19      , 4.01      , 1.6       ,
       1.04      , 1.64      , 1.19      , 4.01      , 1.6       ,
       1.42      , 2.1       , 1.23      , 1.1       , 1.22      ,
       1.5       , 1.36      , 1.32      ])
```

```
df['Display Size'] = df['Display Size'].round(1)
```

```
df['Battery Life'].unique()
```

```
array(['18', '40', '11', '6', '24', '14', '2', '4', '12', '30', '3', '45',
       '5', '10', '48', '7', '16', '9', '25', '72', '60', '56', '70', '1',
       '48 hours', '15', 'Unlimited', '1.5', '20', '8'], dtype=object)
```

```
df['Battery Life'].describe()
```

```
count     379
unique     30
top        14
freq       84
Name: Battery Life, dtype: object
```

```
df['Battery Life'] = df['Battery Life'].replace({'48 hours' : '14', 'Unlimited' : '14'})
```

```
df['Price'] = df['Price'].str[1:]
```

```
df['Water Resistance'] = df['Water Resistance'].astype(float)
```

```
df['Battery Life'] = df['Battery Life'].astype(float)
```

Feature Engineering	Attached the codes in final submission.
Save Processed Data	-