Linear regression is all about dependent and independent variable. It is about prediction, Linear Regression Algorithm is a statistical technique for calculating the value of a dependent variable based on the value of an independent variable.

Example: Let's say you are a coffee owner. You want to predict how much money you will make in a day based on the number of coffee you sell.Here, money is depend on coffee. Money is dependent variable and coffee is independent variable. In this example the dependent variable is the amount of money made in a day (what you are trying to predict). The independent variable is the number of coffee sold (what you are basing the prediction on).

To predict this do few days or week or month survey, To predict this, gather data on the number of coffee sell and how much money earned in several days/week or months.

First step, import libraries and data

In [1]:

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression

#for now lets import this, later we can import more if needed
```

In [2]:

```
#lets import data, just a example, I am importing cupcake data.
df=pd.read_excel(r'C:\Users\USER\Desktop\sample.xlsx')
```

In [3]:

```
print(df)
```

```
          Date  Cupcakes_Sold  Money_Made
0   2023-11-01            200        1000
1   2023-11-02            150         750
2   2023-11-03            175         875
3   2023-11-04            225        1125
4   2023-11-05            250        1250
5   2023-11-06            300        1500
6   2023-11-07            350        1750
7   2023-11-08            275        1375
8   2023-11-09            200        1000
9   2023-11-10            175         875
10  2023-11-11            150         750
11  2023-11-12            225        1125
12  2023-11-13            250        1250
13  2023-11-14            175         875
14  2023-11-15            300        1500
15  2023-11-16            200        1000
16  2023-11-17            225        1125
17  2023-11-18            250        1250
18  2023-11-19            175         875
19  2023-11-20            150         750
20  2023-11-21            200        1000
21  2023-11-22            300        1500
22  2023-11-23            250        1250
23  2023-11-24            175         875
24  2023-11-25            225        1125
25  2023-11-26            200        1000
26  2023-11-27            150         750
27  2023-11-28            175         875
28  2023-11-29            250        1250
29  2023-11-30            300        1500
```

In [4]:

```
df.head(5)
```

Out[4]:

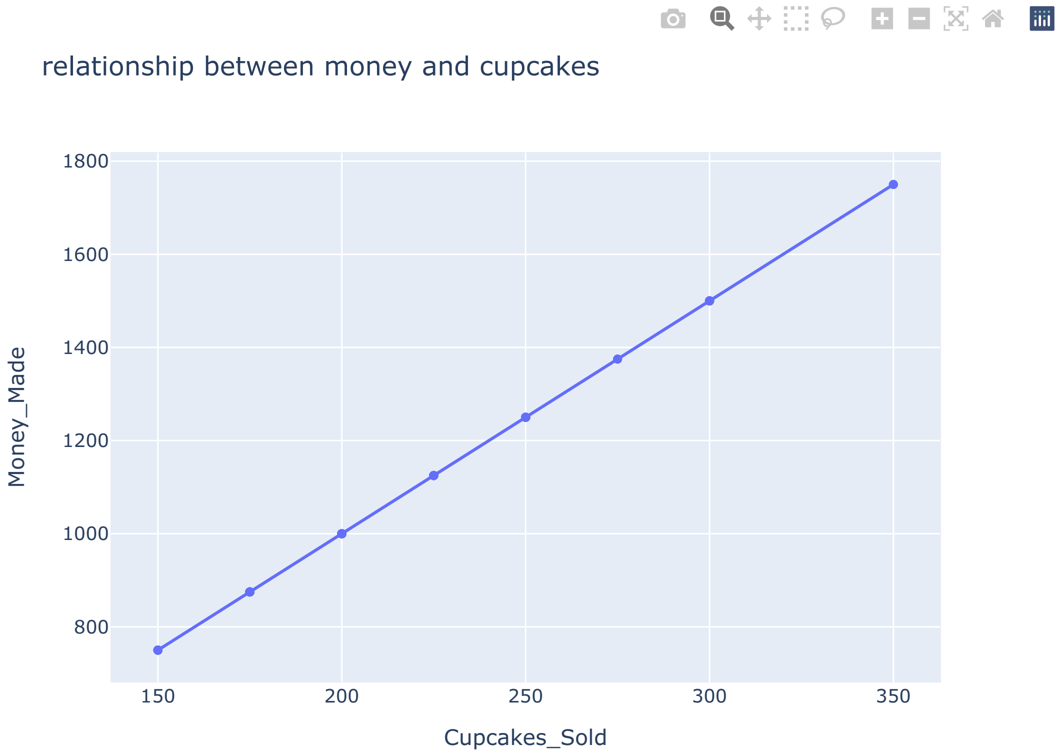
	Date	Cupcakes_Sold	Money_Made
0	2023-11-01	200	1000
1	2023-11-02	150	750
2	2023-11-03	175	875
3	2023-11-04	225	1125
4	2023-11-05	250	1250

In [5]:

```
#now lets look relationship, let look scatter plot
import plotly.express as px
relation_cupcake_money=px.scatter(data_frame=df,x="Cupcakes_Sold",y="Money_Made",
                                  trendline="ols", title="relationship between money and cupcakes")
```

In [6]:

```
relation_cupcake_money.show()
```



Second step (Lets train Machine learning model using linear regression algorthim)

```
In [7]: x1,y1=df["Cupcakes_Sold"],df["Money_Made"]
```

```
In [8]: x,y=np.array(x1).reshape(-1,1),np.array(y1)  #we need to convert independent variable in 2 D array
```

```
In [9]: model = LinearRegression().fit(x, y)
```

```
In [10]: #predict the money made for 25 cupcakes sold
         new_Cupcakes_Sold = [[40]]
         new_Money_Made = model.predict(new_Cupcakes_Sold)
         print("Predicted money made for the cupcakes sold:", new_Money_Made)
```

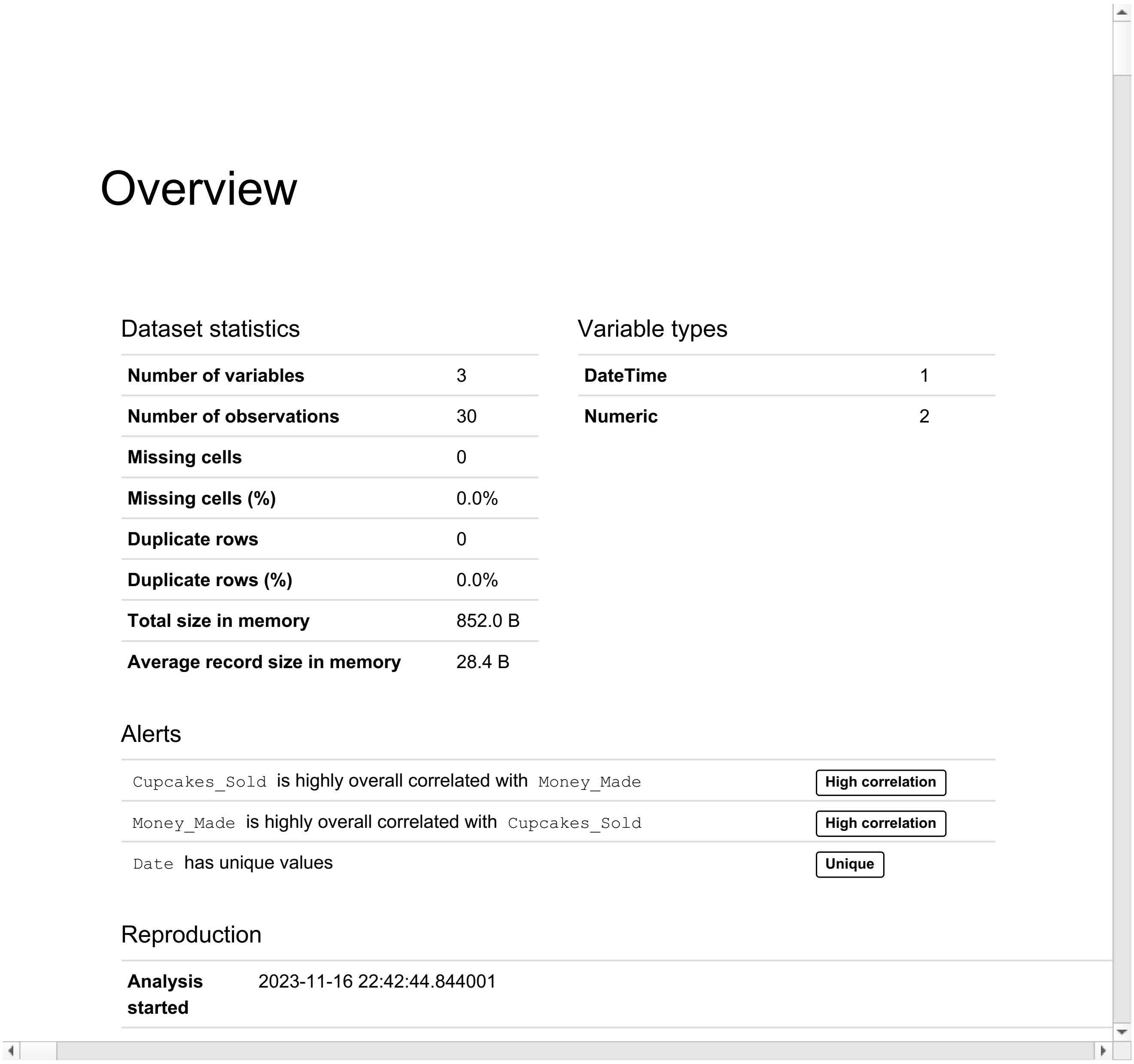
```
Predicted money made for the cupcakes sold: [200.]
```

In this example, the reshape(-1, 1) operation has transformed the one-dimensional array into a two-dimensional array with a single column. This is often useful when working with machine learning algorithms that expect input data in a specific format, such as when using features for linear regression.

JUST TRYING TO SEE FOR SIMPLY ONLY DATA HOW PANADS YDATA PROFILING WORK HERE

```
In [11]: from ydata_profiling import ProfileReport
         ProfileReport(df)
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```



Overview

Dataset statistics

Number of variables	3
Number of observations	30
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	852.0 B
Average record size in memory	28.4 B

Variable types

DateTime	1
Numeric	2

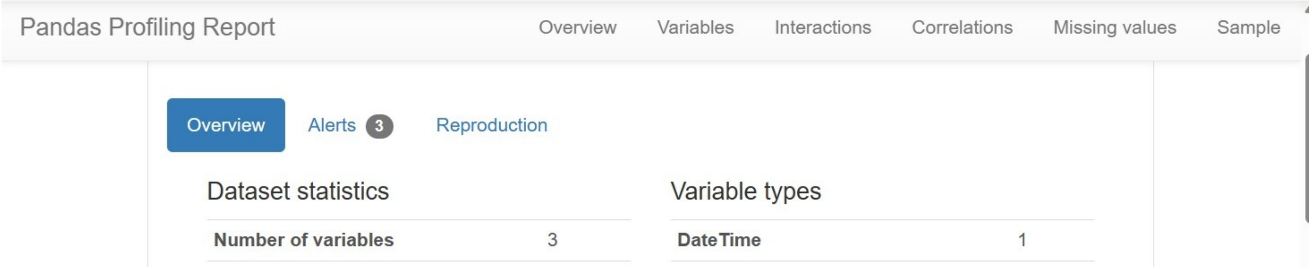
Alerts

`Cupcakes_Sold` is highly overall correlated with `Money_Made`	**High correlation**
`Money_Made` is highly overall correlated with `Cupcakes_Sold`	**High correlation**
`Date` has unique values	**Unique**

Reproduction

Analysis started	2023-11-16 22:42:44.844001

Out[11]:



Dataset statistics

Number of variables	3

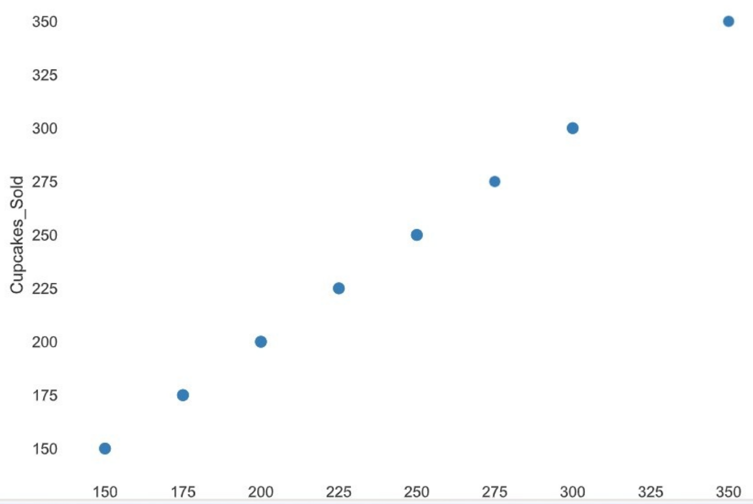
Variable types

DateTime	1

Number of observations	30
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	852.0 B
Average record size in memory	28.4 B

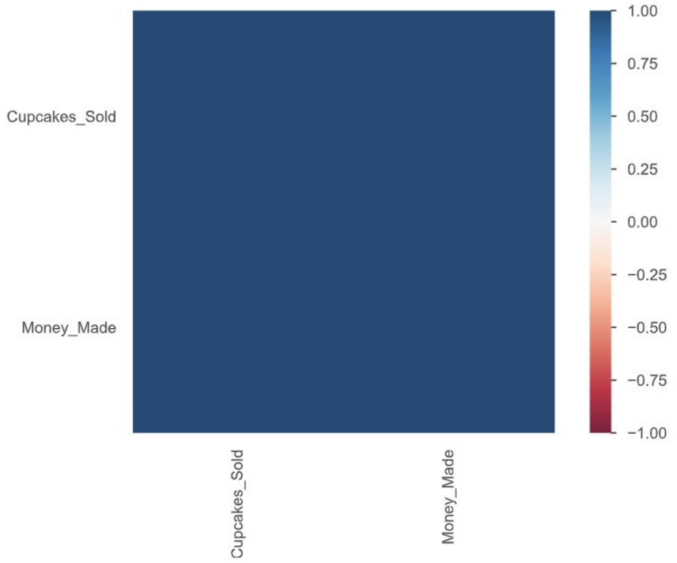
Numeric	2

Pandas Profiling Report Overview Variables Interactions Correlations Missing values Sample



Correlation

Pandas Profiling Report Overview Variables Interactions Correlations Missing values Sample



```
In [ ]:
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js