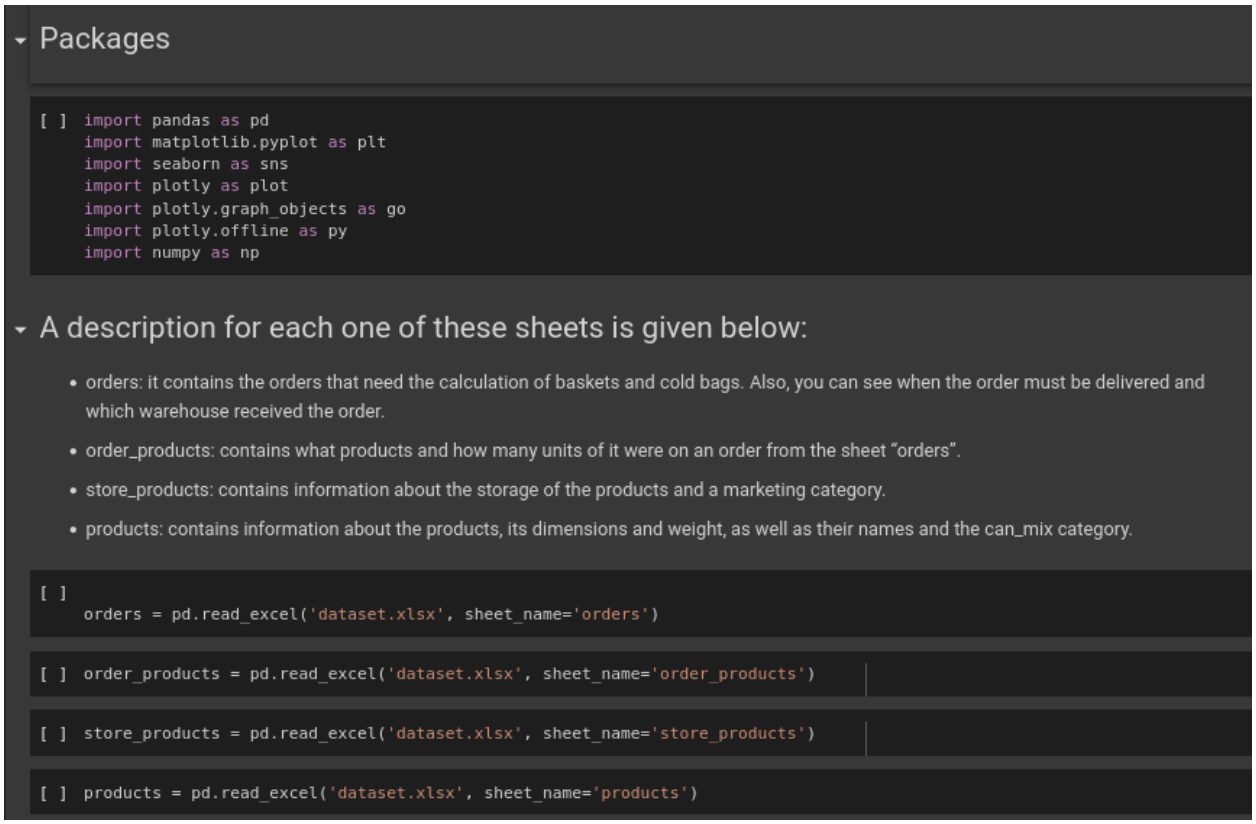
Continuous Improvement

Antonio Vinicius Silva Moura
São Paulo, 2021

This document aims to explain the code of the algorithm that aims to help people in the warehouse to better pack the baskets and also present the results obtained. The code analyzes based on the dataset provided and presents concrete and helpful results.

1/2 - In the first cells we have the import of libraries that we are going to use to make the tool. After that we have the dataset reading so that we can get the information.



3 - As stated in the problem presentation document, the algorithm would have to calculate the volume of baskets and coldbags.

Calculations

$$Totalvolume = height * width * length$$

$$Cubage = Totalvolume * weight$$

The baskets have the following characteristics:

- Length: 50 centimeters
- Width: 40 centimeters
- Height: 60 centimeters

Total weight that it can resit: 25 kilograms

```
[ ]  total_volume_baskets = 50 * 40 * 60
     cubage_baskets = total_volume_baskets  * (25 * 1000)
     cubage_baskets
```

```
3000000000
```

The cold bags have the following characteristics:

- Length: 23 centimeters
- Width: 37 centimeters
- Height: 18 centimeters

Total weight that it can resit: 5 kilograms

```
[ ]  total_volume_coldbag = 23 * 37 * 18
     cubage_coldbag = total_volume_coldbag * (5 * 1000)
     cubage_coldbag
```

```
76590000
```

4 - After that we started our analysis. At first we sorted our information set by their id and after that we got some statistical data like mean, sum, etc.

```
[ ] orders.sort_values(['order_id','warehouse_id'], ascending=False)
    orders.head(3)
```

	order_id	warehouse_id	delivery_date	user_id
0	1781559	1	2020-02-01	89718
1	1781560	1	2020-02-02	487283
2	1781561	5	2020-02-02	477993

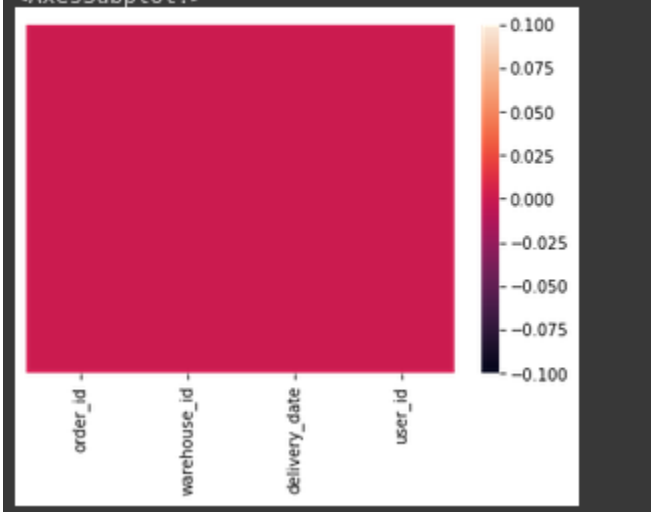
```
[ ] print(orders.shape)
    print()
    print(orders.describe())
    print()
    print(orders.info())
```

```
(26308, 4)

           order_id  warehouse_id        user_id
count  2.630800e+04  26308.000000   26308.000000
mean   1.795241e+06      4.023567  507278.641896
std    7.913503e+03      2.475480  232748.593153
min    1.781559e+06      1.000000      12.000000
25%    1.788343e+06      2.000000  306568.500000
50%    1.795236e+06      5.000000  571123.000000
75%    1.802152e+06      6.000000  720586.000000
max    1.808868e+06    124.000000  766206.000000

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26308 entries, 0 to 26307
Data columns (total 4 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   order_id       26308 non-null  int64
 1   warehouse_id   26308 non-null  int64
 2   delivery_date  26308 non-null  object
 3   user_id        26308 non-null  int64
dtypes: int64(3), object(1)
memory usage: 822.2+ KB
None
```

We now have three heatmap type graphs that show orders, orders product, and store products data. In it we can see the distribution of information and where data is missing.

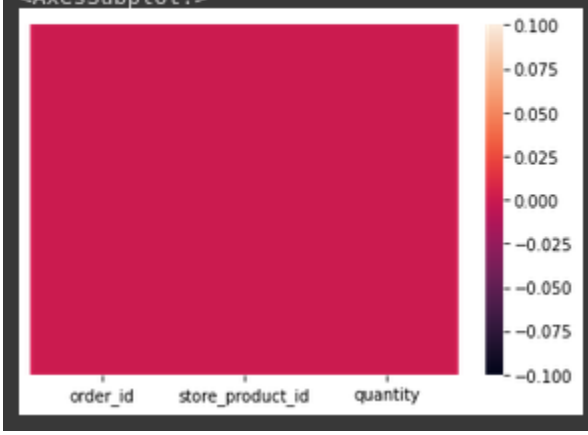
```
sns.heatmap(orders.isnull(),yticklabels=False)
```

```
<AxesSubplot:>
```



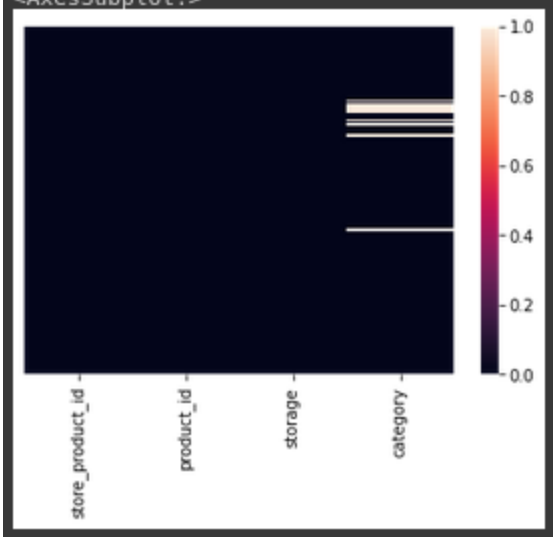

```
sns.heatmap(order_products.isnull(),yticklabels=False)
```

```
<AxesSubplot:>
```



```
sns.heatmap(store_products.isnull(),yticklabels=False)
```

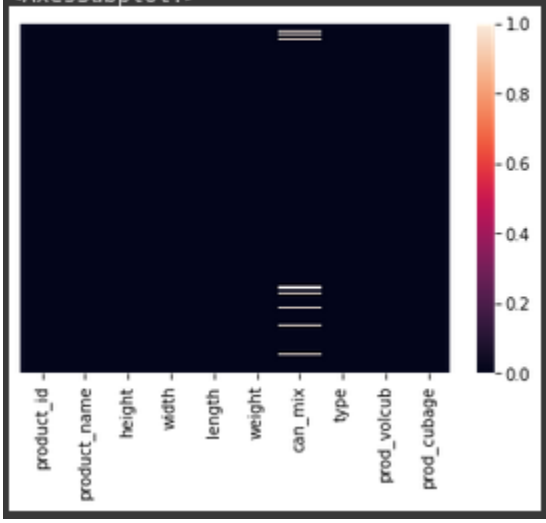
```
<AxesSubplot:>
```



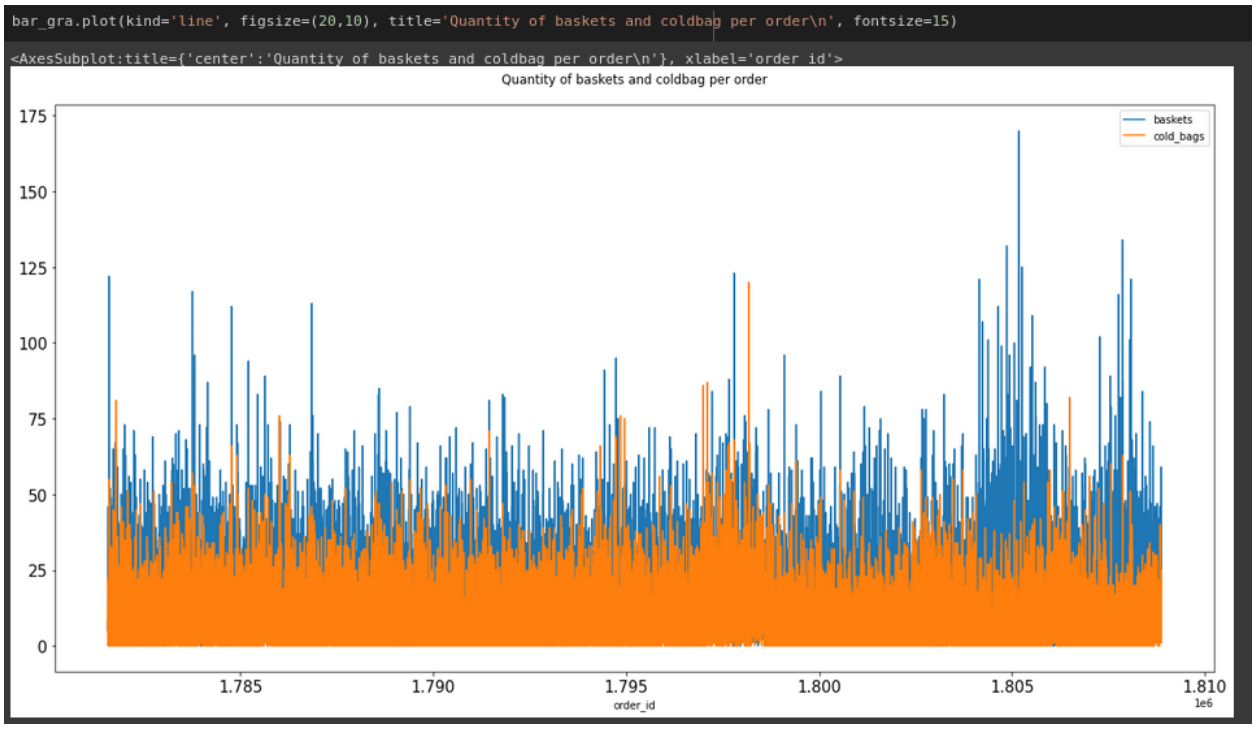
After these charts, we take the name of all the products and build another heatmap chart, and again there we see where information is missing.

```
sns.heatmap(products.isnull(),yticklabels=False)
```

```
<AxesSubplot:>
```

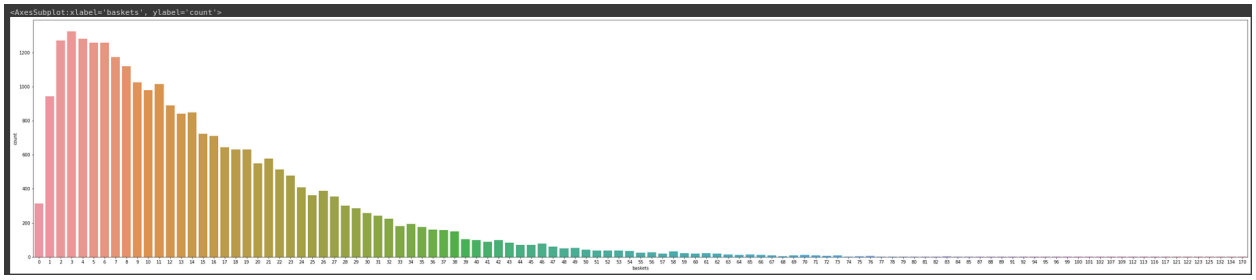


5 - After obtaining all the necessary information, we set up the last graphics that will present the information on the number of baskets and coldbags per order.

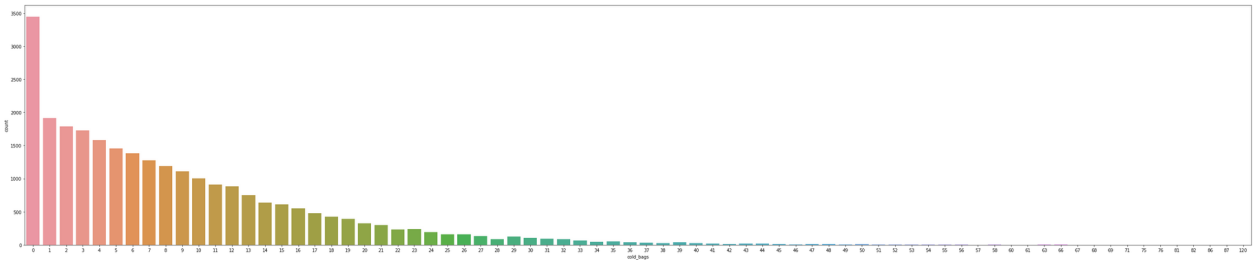


Then, we see the number of baskets and coldbags individually.

baskets:

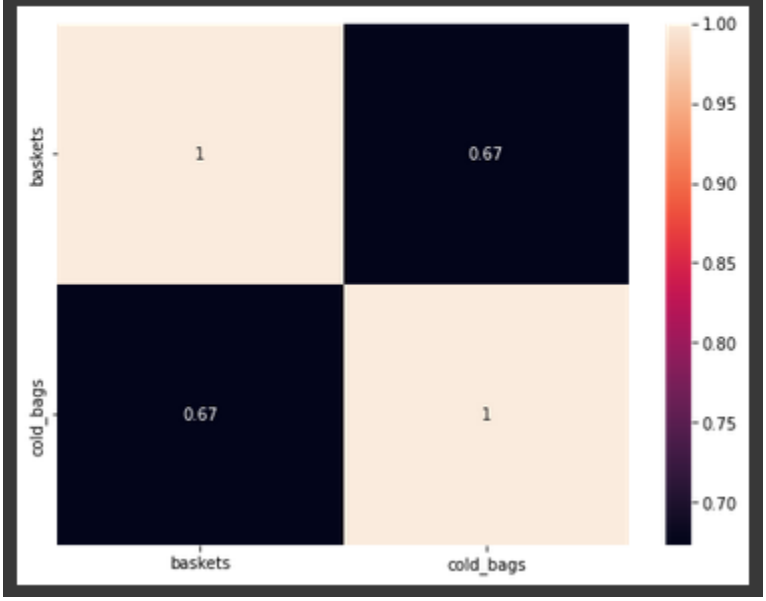


coldbags:



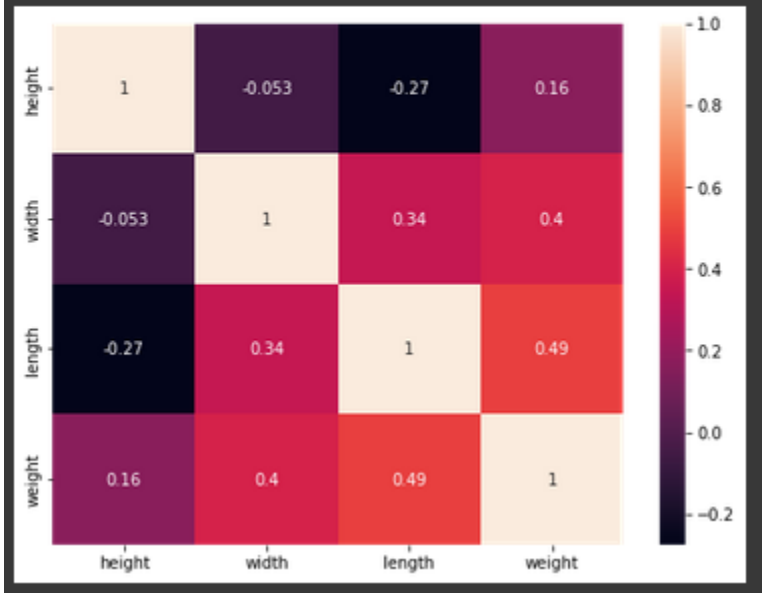
Now we have another heatmap that shows the ratio between baskets and coldbags.

```
corr_df = bar_gra.corr(method='pearson')
plt.figure(figsize=(8, 6))
sns.heatmap(corr_df, annot=True)
plt.show()
```



After that we create a new dataset so that we can analyze the relationship between orders by height, width, length and weight.

```
corr_df = prod.corr(method='pearson')
plt.figure(figsize=(8, 6))
sns.heatmap(corr_df, annot=True)
plt.show()
```



Finally, we have a chart that presents data by date, which starts on February 10th, 2020 and ends on March 12th, 2020.