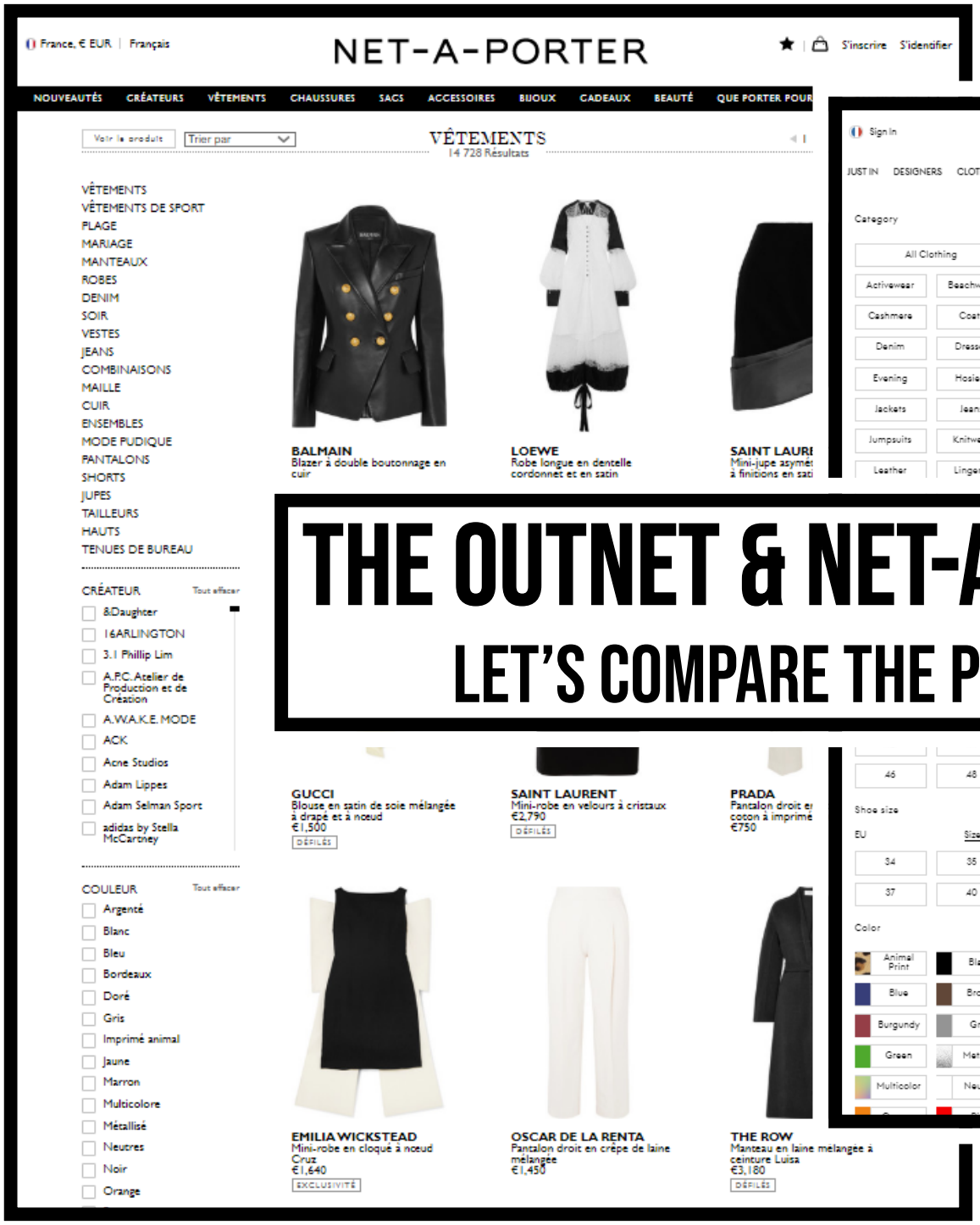
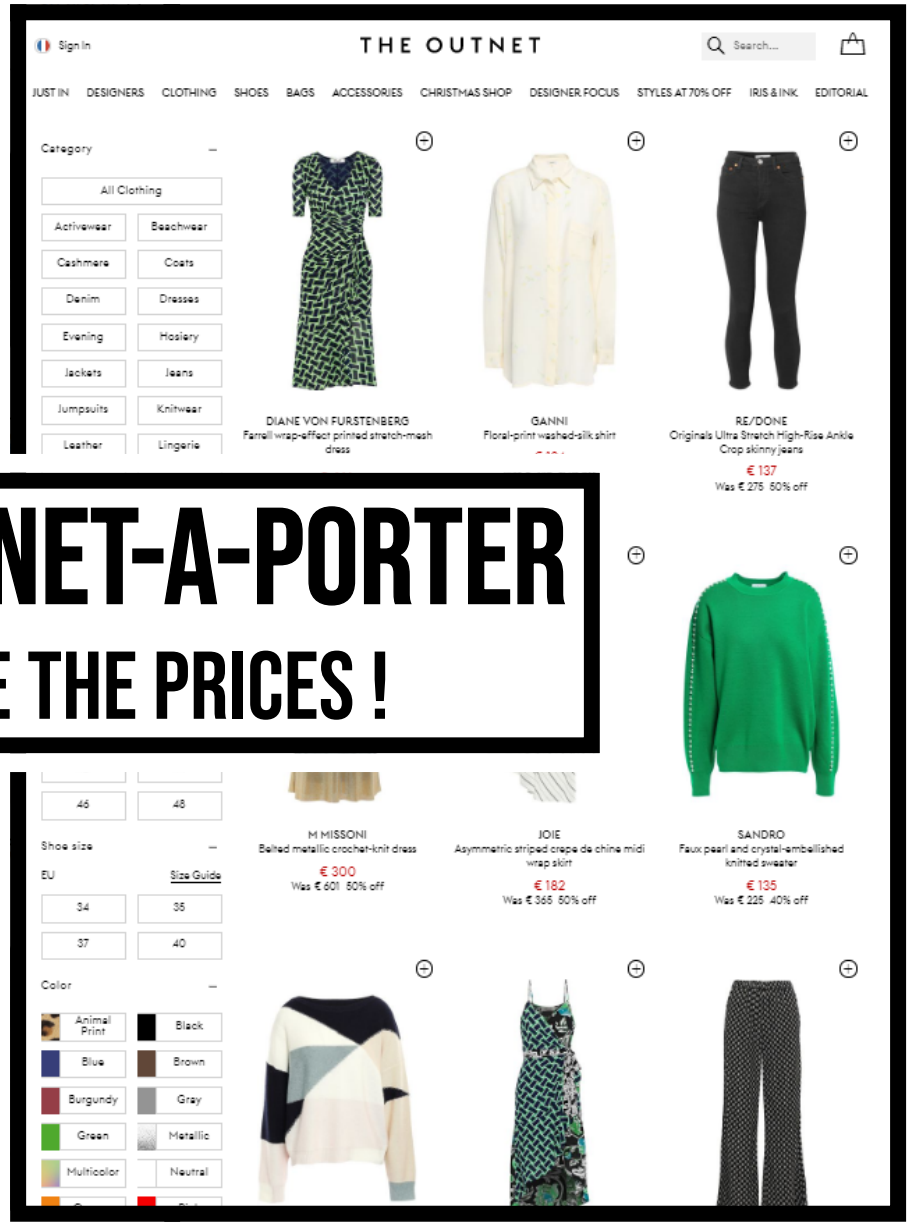



# THE OUTNET & NET-A-PORTER

## LET'S COMPARE THE PRICES !

# COMPARING THE AVERAGE PRICES ON NET-A-PORTER .COM
# VS
# AVERAGE ORIGINAL PRICES ON THEOUTNET.COM

AIM : See if the average original price is higher on the outlet website
--> Do they cheat on the reductions ?

# WEB SCRAPING

## THE OUTNET

SCRAPING

Entrée [1]:
```
import requests as r
from bs4 import BeautifulSoup as BS
import pandas as pd
import re
import time as t
```

```
table_dress=[]
for i in range(1,78):
    if i%2==0:
            t.sleep(2)
    url=f'https://www.theoutnet.com/Search/RenderProductsAsync?ytosQuery=true&linkdepartment=INTL_Dresses_CLOT
    lst=BS(r.get(url).content).select('div.description>span.designer-name,span.discounted>span.value,span.full
    table_url=[(j.text).strip('\r\n\txa').replace('\xa0','') for j in lst]
    table_dress+=table_url
```

Entrée [58]:
```
table_dress
```

Out[58]:
```
['IRIS & INK',
 'Siri wrap-effect satin-crepe dress',
 '145',
 'IRIS & INK',
 'Marne pleated crepe dress',
 '155',
 'IRIS & INK',
 'Elina gathered crepe dress',
 '130',
 'IRIS & INK',
 'Dagny embroidered stretch-knit dress',
 '155',
 'CAMILLA',
 'Crystal-embellished knotted silk crepe de chine maxi dress',
 '387',
 '745',
 '745',
 'TORY BURCH',
```

# WEB SCRAPING

## THE OUTNET

**CLEANING**

Entrée [4]:

```
table_dress_2=[]
for i in range((len(table_dress)-1)):
    if table_dress[i]!=table_dress[i+1]:
        table_dress_2.append(table_dress[i])
table_dress_2.append(table_dress[-1])

table_dress_3=[]
for i in range(0,(len(table_dress_2)-1)):
    if i==0:
        table_dress_3.append(table_dress_2[i])
    elif (table_dress_2[i].isdigit()) & (table_dress_2[i+1].isdigit()==False) & (table_dress_2[i-1].isdigit()==False):
        table_dress_3.append(table_dress_2[i])
        table_dress_3.append('')
    else:
        table_dress_3.append(table_dress_2[i])
table_dress_3.append(table_dress_2[-1])
```

**DISPATCH**

Entrée [8]:

```
Brand=[table_dress_3[i] for i in range(0,len(table_dress_3),4)]
SalePrice=[table_dress_3[i] for i in range(2,len(table_dress_3),4)]
OriginalPrice=[table_dress_3[i] for i in range(3,len(table_dress_3),4)]
Description=[table_dress_3[i] for i in range(1,len(table_dress_3),4)]

colnames=['Brand','Description','SalePrice','OriginalPrice']
```

Entrée [9]:

```
df1=pd.DataFrame([Brand,Description,SalePrice,OriginalPrice], index=colnames).T
df1
```

Out[9]:

| | Brand | Description | SalePrice | OriginalPrice |
|---|---|---|---|---|
| 0 | IRIS & INK | Siri wrap-effect satin-crepe dress | 145 | |
| 1 | IRIS & INK | Marne pleated crepe dress | 155 | |
| 2 | IRIS & INK | Elina gathered crepe dress | 130 | |
| 3 | CAMILLA | Crystal-embellished knotted silk crepe de chin... | 387 | 745 |
| 4 | TORY BURCH | Ruffled striped broderie anglaise cotton midi ... | 280 | 585 |
| 5 | RACHEL ZOE | Asymmetric floral-print silk-georgette dress | 172 | 400 |
| 6 | APIECE APART | Meru asymmetric printed silk crepe de chine wr... | 312 | 600 |
| 7 | TEMPERLEY LONDON | Beaux broderie anglaise-trimmed ruffled Swiss-... | 312 | 695 |
| 8 | GÜL HÜRGEL | Belted gingham cotton and linen-blend maxi shi... | 360 | 720 |

# WEB SCRAPING

## THE OUTNET

SCRAPING

CLEANING

DISPATCH

WEB_SCRAPER

```
def table_generator(url_pattern,pages_to_scrape):
    table=[]
    for i in range(1,pages_to_scrape+1):
        if i%2==0:
            t.sleep(2)
        url=url_pattern%i
        lst=BS(r.get(url).content).select('div.description>span.designer-name,span.discounted>span.value,span.full>span.value,
        table_url=[(j.text).strip('\r\n\txa').replace('\xa0','') for j in lst]
        table+=table_url
    return table

def cleaner(table):
    table_2=[]
    table_3=[]
    for i in range((len(table)-1)):
        if table[i]!=table[i+1]:
            table_2.append(table[i])
    table_2.append(table[-1])
    for i in range(0,(len(table_2)-1)):
        if i==0:
            table_3.append(table_2[i])
        elif (table_2[i].isdigit()) & (table_2[i+1].isdigit()==False) & (table_2[i-1].isdigit()==False):
            table_3.append(table_2[i])
            table_3.append('')
        else:
            table_3.append(table_2[i])
    table_3.append(table_2[-1])
    return table_3

def columns(clean_table):
    Brand=[clean_table[i].upper() for i in range(0,len(clean_table),4)]
    SalePrice=[clean_table[i] for i in range(1,len(clean_table),4)]
    OriginalPrice=[clean_table[i] for i in range(2,len(clean_table),4)]
    Description=[clean_table[i] for i in range(3,len(clean_table),4)]
    return Brand,Description,SalePrice,OriginalPrice

def web_scraper(url_pattern,pages_to_scrape):
    table=table_generator(url_pattern,pages_to_scrape)
    clean_table=cleaner(table)
    Brand,Description,SalePrice,OriginalPrice=columns(clean_table)
    return Brand,Description,SalePrice,OriginalPrice
```

# WEB SCRAPING

## THE OUTNET

IMPORT

```
Entrée [1]: import requests as r
            from bs4 import BeautifulSoup as BS
            import pandas as pd
            import re
            import time as t
            import os
            import TheOutnetWS as WS
```

```
Entrée [2]: os.getcwd()
Out[2]: 'C:\\Users\\mathi\\Documents\\GitHub\\data-labs\\module-1\\lab-web-scraping'
```

```
Entrée [3]: colnames=['Brand','Description','SalePrice','OriginalPrice']
```

APPLY

```
rée [4]: Brand,Description,SalePrice,OriginalPrice=WS.web_scraper('https://www.theoutnet.com/Search/RenderProductsAsync?ytosQuery=true&li
```

DATAFRAME

```
rée [5]: df_dress=pd.DataFrame([Brand,Description,SalePrice,OriginalPrice],index=colnames).T
         df_dress
```

Out[5]:

| | Brand | Description | SalePrice | OriginalPrice |
|---|---|---|---|---|
| 0 | IRIS & INK | Siri wrap-effect satin-crepe dress | 145 | |
| 1 | IRIS & INK | Marne pleated crepe dress | 155 | |
| 2 | IRIS & INK | Elina gathered crepe dress | 130 | |
| 3 | CAMILLA | Crystal-embellished knotted silk crepe de chin... | 387 | 745 |
| 4 | TORY BURCH | Ruffled striped broderie anglaise cotton midi ... | 280 | 585 |
| 5 | RACHEL ZOE | Asymmetric floral-print silk-georgette dress | 172 | 400 |
| 6 | APIECE APART | Meru asymmetric printed silk crepe de chine wr... | 312 | 600 |
| 7 | TEMPERLEY LONDON | Beaux broderie anglaise-trimmed ruffled Swiss-... | 312 | 695 |
| 8 | GÜL HÜRGEL | Belted gingham cotton and linen-blend maxi shi... | 360 | 720 |
| 9 | APIECE APART | Asymmetric off-the-shoulder floral-print cotto... | 235 | 490 |
| 10 | BORGO DE NOR | Gathered floral-print fil coupé maxi dress | 355 | 725 |
| 11 | APIECE APART | Lypie ruffle-trimmed striped cotton-gauze maxi... | 202 | 440 |

# WEB SCRAPING

## THE OUTNET

Entrée [6]:
```
df_dress=df_dress.drop_duplicates()
df_dress
```

DROP DUPLICATES

Entrée [7]:
```
df_dress.Description='Dress'
df_dress
```

DESCRIPTION TO CATEGORY

Out[7]:

| | Brand | Description | SalePrice | OriginalPrice |
|---|---|---|---|---|
| 0 | IRIS & INK | Dress | 145 | |
| 1 | IRIS & INK | Dress | 155 | |
| 2 | IRIS & INK | Dress | 130 | |
| 3 | CAMILLA | Dress | 387 | 745 |
| 4 | TORY BURCH | Dress | 280 | 585 |
| 5 | RACHEL ZOE | Dress | 172 | 400 |
| 6 | APIECE APART | Dress | 312 | 600 |

Entrée [15]:
```
df_skirt=df_skirt.drop((df_skirt[df_skirt.OriginalPrice==''].index.tolist()),axis=0)
df_skirt
```

Out[15]:

| | Brand | Description | SalePrice | OriginalPrice |
|---|---|---|---|---|
| 1 | APIECE APART | Skirt | 157 | 350 |
| 2 | ALICE + OLIVIA | Skirt | 305 | 611 |
| 3 | RE/DONE | Skirt | 159 | 290 |
| 4 | APIECE APART | Skirt | 199 | 370 |
| 5 | MERLETTE | Skirt | 167 | 390 |
| 6 | RACHEL ZOE | Skirt | 178 | 356 |

DROP ROWS WITHOUT ORIGINAL PRICE

# WEB SCRAPING

## THE OUTNET

Entrée [14]:
```
def mode_agg(x):
    return(''.join(x.mode().values.tolist()))
```

Entrée [16]:
```
df_skirt.OriginalPrice=df_skirt.OriginalPrice.astype('int')
df_skirt.SalePrice=df_skirt.SalePrice.astype('int')
df_skirt1=df_skirt.groupby('Brand').agg({'Description':mode_agg,'SalePrice':'mean','OriginalPrice':'mean'}).reset_index().copy()
df_skirt1
```

Out[16]:

| | Brand | Description | SalePrice | OriginalPrice |
|---|---|---|---|---|
| 0 | 3.1 PHILLIP LIM | Skirt | 284.833333 | 623.333333 |
| 1 | 3x1 | Skirt | 149.333333 | 299.000000 |
| 2 | 7 FOR ALL MANKIND | Skirt | 105.666667 | 263.333333 |
| 3 | A.L.C. | Skirt | 200.125000 | 454.375000 |
| 4 | A.P.C. | Skirt | 95.000000 | 175.000000 |
| 5 | A.W.A.K.E. | Skirt | 245.000000 | 490.000000 |
| 6 | ACNE STUDIOS | Skirt | 164.444444 | 350.555556 |
| 7 | ADAM LIPPES | Skirt | 351.666667 | 898.250000 |
| 8 | ADEAM | Skirt | 251.000000 | 535.000000 |
| 9 | ADIDAS ORIGINALS | Skirt | 20.000000 | 80.000000 |
| 10 | AGNONA | Skirt | 510.555556 | 1231.555556 |
| 11 | ALAÏA | Skirt | 882.000000 | 2716.666667 |

GROUP BY BRAND

# WEB SCRAPING

## NET-À-PORTER

Entrée [2]:
```
headers={'accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-excl
         'accept-encoding': 'gzip, deflate, br',
         'accept-language': 'fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7',
         'cache-control': 'max-age=0',
         'cookie': '_qubitTracker=hei2mrvo6e8-0k2d6y6d4-jdr2t40; country_iso=FR; channel=intl; lang_iso=fr; geoIP=FR; deviceType=
         'referer': 'https://www.net-a-porter.com/fr/fr/?&cm_mmc=GoogleFR%E2%80%94c-_-NAP_FR_FR-_-Brand-_-Net+A+Porter_Exact-_-n
         'sec-fetch-mode': 'navigate',
         'sec-fetch-site': 'same-origin',
         'sec-fetch-user': '?1',
         'upgrade-insecure-requests': '1',
         'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/77.0.3865.120 Sa
```

SCRAPING

```
table_brand=[]
for i in range(1,68):
    if i%2==0:
            t.sleep(2)
    url=f'https://www.net-a-porter.com/fr/fr/d/Shop/Clothing/Dresses?cm_sp=topnav-_-clothing-_-dresses&pn=%i&npp=60&image_view=p
    lst=BS(r.get(url,headers=headers).content).select('a>span.designer')
    table_url=[(i.text).upper() for i in lst]
    table_brand+=table_url
table_brand
```

```
 'PROENZA SCHOULER',
 'PUSHBUTTON',
 'GIAMBATTISTA VALLI',
 'HOLZWEILER',
 'MONSE',
 'JACQUEMUS',
 'PRADA',
 'REJINA PYO',
 'LES RÊVERIES',
 'STELLA MCCARTNEY',
 'VALENTINO',
```

# WEB SCRAPING

## NET-À-PORTER

Entrée [4]:
```
table_price=[]
for i in range(1,68):
    if i%2==0:
            t.sleep(3)
    url='https://www.net-a-porter.com/fr/fr/d/Shop/Clothing/Dresses?cm_sp=topnav-_-clothing-_-dresses&pn=%i&npp=60&image_view=pr
    lst=BS(r.get(url=url,headers=headers).content).select('span.price')
    table_url=[(i.text).strip('\n\t€').replace(',','') for i in lst]
    table_price=table_price+table_url
table_price
```

Out[4]:
```
['3495',
 '1195',
 '605',
 '1095',
 '2890',
 '2290',
 '3630',
 '1727',
```

Entrée [5]:
```
table_desc=[]
for i in range(1,68):
    if i%2==0:
            t.sleep(3)
    url='https://www.net-a-porter.com/fr/fr/d/Shop/Clothing/Dresses?cm_sp=topnav-_-clothing-_-dresses&pn='+str(i)+'&npp=60&image_
    lst=BS(r.get(url,headers=headers).content).select('div.description>a')
    table_url=[''.join(re.findall(r'title=.*',str(j))).strip('">\"[]').replace('&amp;','&').replace('title=','').replace('"','')
    table_desc+=table_url
table_desc
```

Out[5]:
```
['Robe midi en crêpe de laine mélangée à imprimé fleuri et à plis surpiqués',
 'Robe en crêpe à finitions en cuir synthétique',
 'Robe longue en jersey stretch à imprimé fleuri Jil',
 'Robe midi en toile de coton à ceinture',
 'Robe midi bicolore en mailles côtelées',
 'Robe en cady à finitions en cuir et à ceinture ',
```

# WEB SCRAPING

## NET-À-PORTER

**FUNCTION**

```
def table_generator(url_pattern,headers,pages_to_scrape):
    table_brand=[]
    table_price=[]
    table_desc=[]
    for i in range(1,pages_to_scrape+1):
        if i%2==0:
            t.sleep(2)
        url=url_pattern%i
        soup=BS(r.get(url,headers=headers).content)
        lst1=soup.select('a>span.designer')
        table_url1=[(j.text).upper() for j in lst1]
        table_brand+=table_url1
        lst2=soup.select('span.price')
        table_url2=[(k.text).strip('\n\t€').replace(',','') for k in lst2]
        table_price+=table_url2
        lst3=soup.select('div.description>a')
        table_url3=[''.join(re.findall(r'title=.*',str(l))).strip('">\"[]').replace('&amp;','&').replace('title=','').replace('"
        table_desc+=table_url3


    return table_brand, table_desc, table_price
```

**APPLY**

```
Brand1,Description1,OriginalPrice1=table_generator(f'https://www.net-a-porter.com/fr/fr/d/Shop/Clothing/Pants?cm_sp=topnav-_-clo
df_pants=pd.DataFrame([Brand1,Description1,OriginalPrice1], index=colnames).T
df_pants=df_pants.drop_duplicates()
df_pants.Description='Pants'
df_pants
```

Out[12]:

| | Brand | Description | OriginalPrice |
|---|---|---|---|
| 0 | ROLAND MOURET | Pants | 530 |
| 1 | MARA HOFFMAN | Pants | 350 |
| 2 | LA COLLECTION | Pants | 485 |
| 3 | ENVELOPE1976 | Pants | 300 |
| 4 | STELLA MCCARTNEY | Pants | 535 |

# WEB SCRAPING

## SAVING TO CSV

### THE OUTNET FILES

```
df_dress1.to_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\DressTON.csv',index=False)
df_pants1.to_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\PantsTON.csv',index=False)
df_jacket1.to_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\JacketTON.csv',index=False)
df_top1.to_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\TopTON.csv',index=False)
df_skirt1.to_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\SkirtTON.csv',index=False)
```

### NET-À-PORTER FILES

```
df_dress1.to_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\DressNAP.csv',index=False)
```

```
df_pants1.to_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\PantsNAP.csv',index=False)
```

```
df_jacket1.to_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\JacketNAP.csv',index=False)
```

```
df_top1.to_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\TopNAP.csv',index=False)
```

```
df_skirts1.to_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\SkirtNAP.csv',index=False)
```

# DATA CLEANING

## AGGREGATE ALL DATASETS TO ONE

Entrée [52]:
```
NAP=pd.DataFrame()
NAP=NAP.append(Skirt_NAP).append(Dress_NAP).append(Top_NAP).append(Jacket_NAP).append(Pants_NAP)
NAP=NAP.reset_index().drop('index',axis=1)
NAP.OriginalPrice=NAP.OriginalPrice.round(2)
NAP
```

| | | | |
|---|---|---|---|
| **21** | ALICE + OLIVIA | Skirt | 514.17 |
| **22** | ALLUDE | Skirt | 459.00 |
| **23** | ALTUZARRA | Skirt | 958.33 |
| **24** | AMIRI | Skirt | 1954.00 |
| **25** | ANDERSSON BELL | Skirt | 382.67 |
| **26** | ANINE BING | Skirt | 259.86 |
| **27** | ANN DEMEULEMEESTER | Skirt | 1108.00 |
| **28** | ANNA MASON | Skirt | 603.67 |
| **29** | ANNA QUAN | Skirt | 390.00 |

Entrée [63]:
```
Data=TON.append(NAP).reset_index().drop('index',axis=1).copy()
Data
```

Out[63]:

| | Brand | Category | OriginalPrice | SalePrice |
|---|---|---|---|---|
| **0** | 3.1 PHILLIP LIM | Skirt | 623.33 | 284.83 |
| **1** | 3x1 | Skirt | 299.00 | 149.33 |
| **2** | 7 FOR ALL MANKIND | Skirt | 263.33 | 105.67 |
| **3** | A.L.C. | Skirt | 454.38 | 200.12 |
| **4** | A.P.C. | Skirt | 175.00 | 95.00 |
| **5** | A.W.A.K.E. | Skirt | 490.00 | 245.00 |
| **6** | ACNE STUDIOS | Skirt | 350.56 | 164.44 |
| **7** | ADAM LIPPES | Skirt | 898.25 | 351.67 |
| **8** | ADEAM | Skirt | 535.00 | 251.00 |
| **9** | ADIDAS ORIGINALS | Skirt | 80.00 | 20.00 |

# DATA CLEANING

## KEEP ONLY DUPLICATES FROM COLUMNS BRAND & CATEGORY

Entrée [64]:
```
Data1=Data[Data.duplicated(subset=['Brand','Category'],keep=False)].copy()
Data1=Data1.reset_index().drop('index',axis=1)
Data1
```

Out[64]:

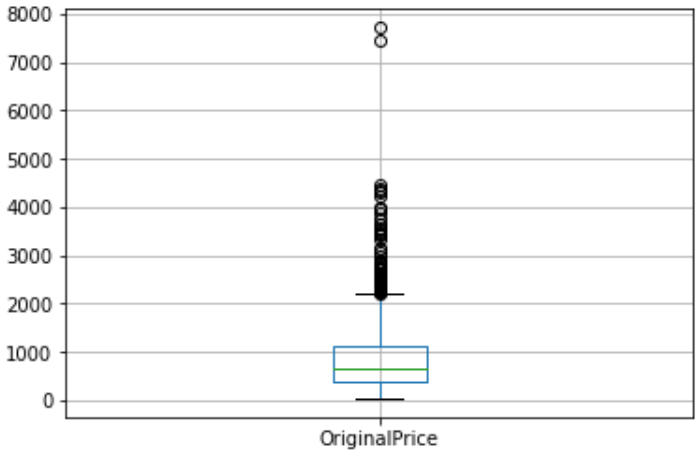
| | Brand | Category | OriginalPrice | SalePrice |
|---|---|---|---|---|
| 0 | 3.1 PHILLIP LIM | Skirt | 623.33 | 284.83 |
| 1 | ACNE STUDIOS | Skirt | 350.56 | 164.44 |
| 2 | ADAM LIPPES | Skirt | 898.25 | 351.67 |
| 3 | ADEAM | Skirt | 535.00 | 251.00 |
| 4 | AGNONA | Skirt | 1231.56 | 510.56 |
| 5 | ALAÏA | Skirt | 2716.67 | 882.00 |
| 6 | ALEXACHUNG | Skirt | 310.83 | 164.33 |
| 7 | ALEXANDER MCQUEEN | Skirt | 2579.60 | 1174.00 |
| 8 | ALEXANDER WANG | Skirt | 550.00 | 263.75 |
| 9 | ALEXANDERWANG.T | Skirt | 340.00 | 157.71 |
| 10 | ALICE + OLIVIA | Skirt | 618.71 | 283.35 |
| 11 | ALTUZARRA | Skirt | 962.92 | 450.50 |

# DATA CLEANING

## OUTLIERS

Entrée [18]:
```
Data1.boxplot(column='OriginalPrice')
```

Out[18]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2238843f0f0>
```




Entrée [19]:
```
Data1[Data1.OriginalPrice>7000]
```

Out[19]:

| | Brand | Category | OriginalPrice | SalePrice |
|---|---|---|---|---|
| **361** | BALMAIN | Jacket | 7713.0 | 3925.8 |
| **749** | ALEXANDER MCQUEEN | Dress | 7440.0 | NaN |

Entrée [20]:
```
Data1.shape
```

Out[20]:
```
(1232, 4)
```

Entrée [21]:
```
Data1=Data1.drop(Data1[(Data1.Brand=='BALMAIN')&(Data1.Category=='Jacket')].index)
Data1=Data1.drop(Data1[(Data1.Brand=='ALEXANDER MCQUEEN')&(Data1.Category=='Dress')].index)
Data1.shape
```

Out[21]:
```
(1228, 4)
```

# DATA CLEANING

## SEPARATE DATAFRAMES AND SAVING THEM

Entrée [22]:
```
DataNAP=Data1[Data1.duplicated(subset=['Brand','Category'],keep='first')].copy()
```

Entrée [23]:
```
DataTON=Data1[Data1.duplicated(subset=['Brand','Category'],keep='last')].copy()
```

Entrée [24]:
```
DataTON=DataTON.sort_values(by='Brand').reset_index().drop('index',axis=1)
DataTON
```

Out[24]:

| | Brand | Category | OriginalPrice | SalePrice |
|---|---|---|---|---|
| 0 | 3.1 PHILLIP LIM | Skirt | 623.33 | 284.83 |
| 1 | 3.1 PHILLIP LIM | Jacket | 763.44 | 320.25 |
| 2 | 3.1 PHILLIP LIM | Top | 415.88 | 173.21 |
| 3 | 3.1 PHILLIP LIM | Pants | 577.19 | 254.19 |
| 4 | ACNE STUDIOS | Skirt | 350.56 | 164.44 |
| 5 | ACNE STUDIOS | Top | 332.50 | 155.30 |
| 6 | ACNE STUDIOS | Jacket | 1037.50 | 518.75 |
| 7 | ACNE STUDIOS | Pants | 475.50 | 219.20 |
| 8 | ADAM LIPPES | Skirt | 898.25 | 351.67 |
| 9 | ADAM LIPPES | Top | 644.13 | 289.87 |
| 10 | ADAM LIPPES | Pants | 815.71 | 337.52 |

Entrée [25]:
```
DataNAP=DataNAP.sort_values(by='Brand').reset_index().drop('index',axis=1)
DataNAP
```

Out[25]:

| | Brand | Category | OriginalPrice | SalePrice |
|---|---|---|---|---|
| 0 | 3.1 PHILLIP LIM | Skirt | 552.88 | NaN |
| 1 | 3.1 PHILLIP LIM | Jacket | 759.75 | NaN |
| 2 | 3.1 PHILLIP LIM | Top | 503.08 | NaN |
| 3 | 3.1 PHILLIP LIM | Pants | 564.25 | NaN |
| 4 | ACNE STUDIOS | Skirt | 390.00 | NaN |
| 5 | ACNE STUDIOS | Top | 242.94 | NaN |
| 6 | ACNE STUDIOS | Jacket | 1476.43 | NaN |
| 7 | ACNE STUDIOS | Pants | 765.56 | NaN |
| 8 | ADAM LIPPES | Skirt | 1842.50 | NaN |
| 9 | ADAM LIPPES | Top | 732.78 | NaN |
| 10 | ADAM LIPPES | Pants | 780.00 | NaN |

Entrée [26]:
```
DataNAP.to_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\DataNAP.csv',index=False)
DataTON.to_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\DataTON.csv',index=False)
```

# ANALYSE
## & CONCLUSION

Entrée [27]:
```
Compare=pd.DataFrame()
for i in range(DataNAP.shape[0]):
    if (DataTON.OriginalPrice[i])>(DataNAP.OriginalPrice[i]*1.3):
        Compare=Compare.append(pd.DataFrame(DataNAP[['Brand','Category']].loc[i]).T)
Compare.shape
```

Out[27]: (113, 2)

Entrée [28]:
```
Compare=pd.DataFrame()
for i in range(DataNAP.shape[0]):
    if (DataTON.OriginalPrice[i])>(DataNAP.OriginalPrice[i]*1.5):
        Compare=Compare.append(pd.DataFrame(DataNAP[['Brand','Category']].loc[i]).T)
Compare.shape
```

Out[28]: (59, 2)

Entrée [29]:
```
Compare=pd.DataFrame()
for i in range(DataNAP.shape[0]):
    if (DataTON.OriginalPrice[i])>(DataNAP.OriginalPrice[i]):
        Compare=Compare.append(pd.DataFrame(DataNAP[['Brand','Category']].loc[i]).T)
Compare.shape
```

Out[29]: (302, 2)

Entrée [30]:
```
Compare=pd.DataFrame()
for i in range(DataNAP.shape[0]):
    if (DataTON.OriginalPrice[i])>(DataNAP.OriginalPrice[i]*2):
        Compare=Compare.append(pd.DataFrame(DataNAP[['Brand','Category']].loc[i]).T)
Compare.shape
```

Out[30]: (12, 2)

# ANALYSE
## & CONCLUSION

48% of average prices are higher on The Outnet then on Net-à-porter

18% prices are more then 30% more expensive on The Outnet vs Net-à-porter

The average price on Net-à-porter is 1% higher then on The Outnet

**NOT CHEATING !**

# PIPELINE

```
DataNAP=pd.read_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\DataNAP.csv')
DataTON=pd.read_csv(r'C:\Users\mathi\Documents\GitHub\data-labs\module-1\lab-web-scraping\DataTON.csv')


def check():
    Brand=input("Brand ?")
    if Brand=='Q':
        print('Bye Bye')
        return None
    Category=input("Category ?")
    if Category=='Q':
        print('Bye Bye')
        return None
    elif (Brand not in DataTON.Brand.tolist()) or (Category not in DataNAP.Category.tolist()):
        print('non valid choice')
        return check()
    a=DataTON.OriginalPrice[(DataTON.OriginalPrice[(DataTON.Brand==Brand)&(DataTON.Category==Category)].index[0])]
    b=DataNAP.OriginalPrice[(DataNAP.OriginalPrice[(DataNAP.Brand==Brand)&(DataNAP.Category==Category)].index[0])]
    if a > b*1.2:
        print('cheateeeer')
        return check()
    else :
        print("thats's ok")
        return check()

print('Hello,\nWelcome on the comparator.\nPlease enter :')
check()
```