Homework 1

1. New York Times (50 points, Adapted from Rachel Schutt at Columbia) There are 31 datasets named nyt1.csv, nyt2.csv,...,nyt31.csv, which can be found on my webpage. Each one represents one day's worth of ad impressions and clicks on the New York Times homepage in May, 2012 (these are simulated). Each row represents a single user. There are 5 columns: age, gender (0=female, 1=male), number of impressions (page views), number clicks (actions) and whether the user was logged.in.

```
import pandas as pd
import matplotlib.pyplot as plt
import os
import numpy as np
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

```
%cd /content/drive/MyDrive/808W Assignments/HW1/Dataset1/
```

```
/content/drive/MyDrive/808W Assignments/HW1/Datasets
```

```
files = [file for file in os.listdir()]
```

```
df = pd.concat(map(pd.read_csv, files), ignore_index=True)
```

```
df.dropna(inplace=True)
print(df)
```

```
               Age  Gender  Impressions  Clicks  Signed_In
0               36       0            3       0          1
1               73       1            3       0          1
2               30       0            3       0          1
3               49       1            3       0          1
4               47       1           11       0          1
...            ...     ...          ...     ...        ...
14905860        41       1            4       0          1
14905861         0       0            5       0          0
14905862        22       1            3       0          1
14905863        59       0            5       1          1
14905864        29       1            4       0          1

[14905865 rows x 5 columns]
```

```
#df.to_csv('/content/drive/MyDrive/808W Assignments/HW1/'"alldata.csv")
```

a) Create a new variable, age_group, that categorizes users as "<18", "18-24", "2534", "35-44", "45-54", "55-64" and "65+".

```
age_group = [] #required variable
```

```
for row in df.itertuples(index=True, name='Pandas'):
  age = getattr(row, "Age")
  if age<18:
    age_group.append("<18")
  elif age >= 18 and age <= 24:
    age_group.append("18-24")
  elif age >= 25 and age <= 34:
    age_group.append("25-34")
  elif age >= 35 and age <= 44:
    age_group.append("35-44")
  elif age >= 45 and age <= 54:
    age_group.append("45-54")
  elif age >= 55 and age <= 64:
    age_group.append("55-64")
  elif age >= 65:
    age_group.append("65+")
  else:
    print("invalid age")
print(len(age_group)) # corresponds to the number of data points
```

```
14905865
```

```
df['Age Group'] = age_group
print(df) # new variable that classifies into age group is added to the original document.
```

```
               Age  Gender  Impressions  Clicks  Signed_In Age Group
0               36       0            3       0          1     35-44
1               73       1            3       0          1       65+
2               30       0            3       0          1     25-34
3               49       1            3       0          1     45-54
4               47       1           11       0          1     45-54
...            ...     ...          ...     ...        ...       ...
14905860        41       1            4       0          1     35-44
14905861         0       0            5       0          0       <18
14905862        22       1            3       0          1     18-24
14905863        59       0            5       1          1     55-64
14905864        29       1            4       0          1     25-34

[14905865 rows x 6 columns]
```

b. For a single day

i) Plot the distributions of number impressions and click-through-rate (CTR=# clicks/#

impressions), for these 6 age categories. [You will turn in a .R and .html file where the latter will show this plot]

```
ctr = []
for row in df.itertuples(index=True, name='Pandas'):
  if getattr(row, "Impressions") == 0:
    ctr.append(0)
  else:
    ctr.append(round(getattr(row,"Clicks")/getattr(row, "Impressions"),1))
```

```
df['CTR'] = ctr
print(df) # ctr variable is added to the original document.
```

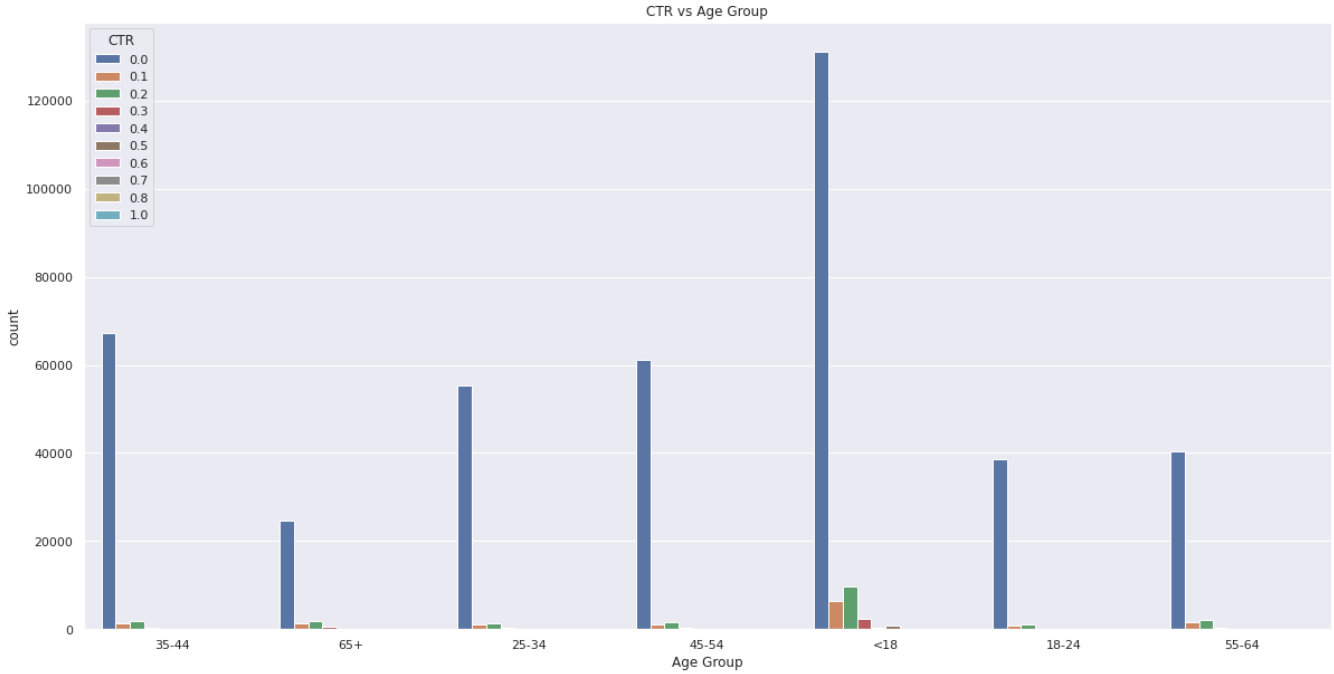
```
               Age  Gender  Impressions  Clicks  Signed_In Age Group  CTR
     0          36       0            3       0          1     35-44  0.0
     1          73       1            3       0          1       65+  0.0
     2          30       0            3       0          1     25-34  0.0
     3          49       1            3       0          1     45-54  0.0
     4          47       1           11       0          1     45-54  0.0
     ...       ...     ...          ...     ...        ...       ...  ...
     14905860   41       1            4       0          1     35-44  0.0
     14905861    0       0            5       0          0       <18  0.0
     14905862   22       1            3       0          1     18-24  0.0
     14905863   59       0            5       1          1     55-64  0.2
     14905864   29       1            4       0          1     25-34  0.0

     [14905865 rows x 7 columns]
```

```
df1b = df.loc[0:458440]
print(df1b)
```

```
             Age  Gender  Impressions  Clicks  Signed_In Age Group  CTR
     0        36       0            3       0          1     35-44  0.0
     1        73       1            3       0          1       65+  0.0
     2        30       0            3       0          1     25-34  0.0
     3        49       1            3       0          1     45-54  0.0
     4        47       1           11       0          1     45-54  0.0
     ...     ...     ...          ...     ...        ...       ...  ...
     458436    0       0            2       0          0       <18  0.0
     458437    0       0            4       0          0       <18  0.0
     458438   72       1            5       0          1       65+  0.0
     458439    0       0            5       0          0       <18  0.0
     458440    0       0            3       0          0       <18  0.0

     [458441 rows x 7 columns]
```

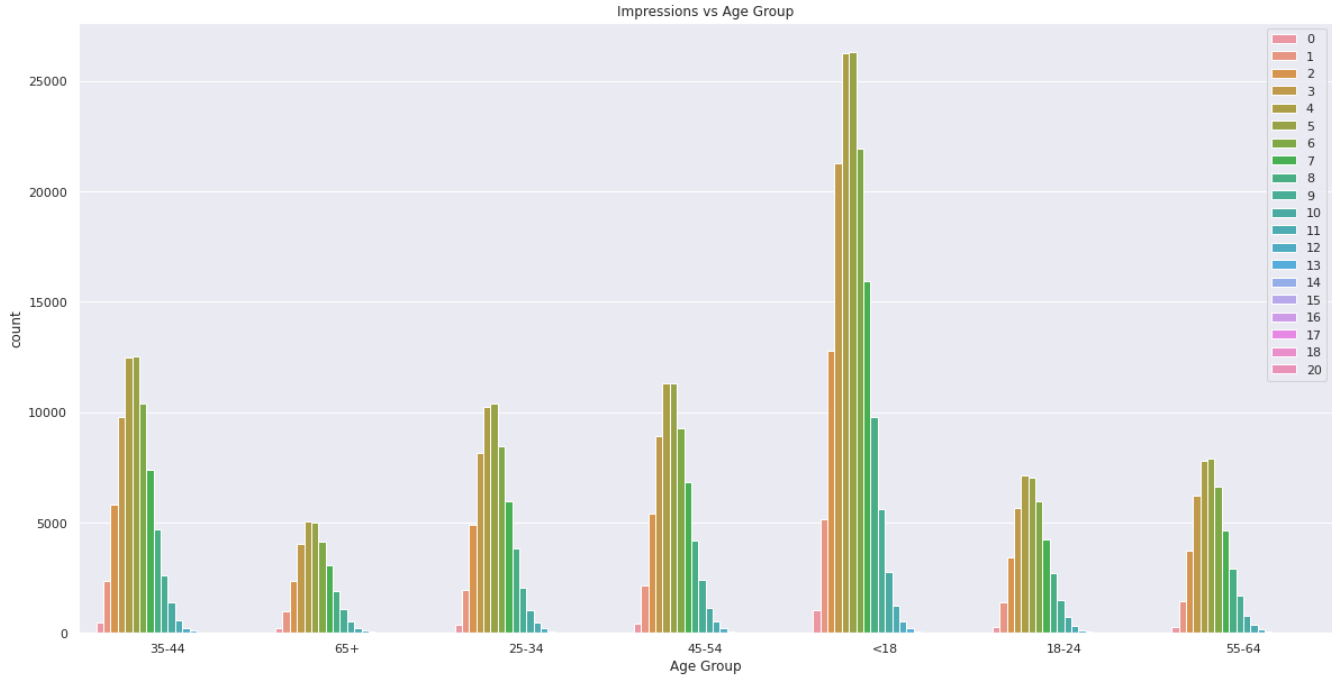
```
sns.set_theme()
sns.set(rc={'figure.figsize':(20,10)})
sns.countplot(x='Age Group', hue='CTR', data=df1b)
plt.title("CTR vs Age Group")
```

```
Text(0.5, 1.0, 'CTR vs Age Group')
```




```
sns.countplot(x='Age Group', hue='Impressions', data=df1b)
plt.legend(loc='upper right')
plt.title("Impressions vs Age Group")
```

```
Text(0.5, 1.0, 'Impressions vs Age Group')
```

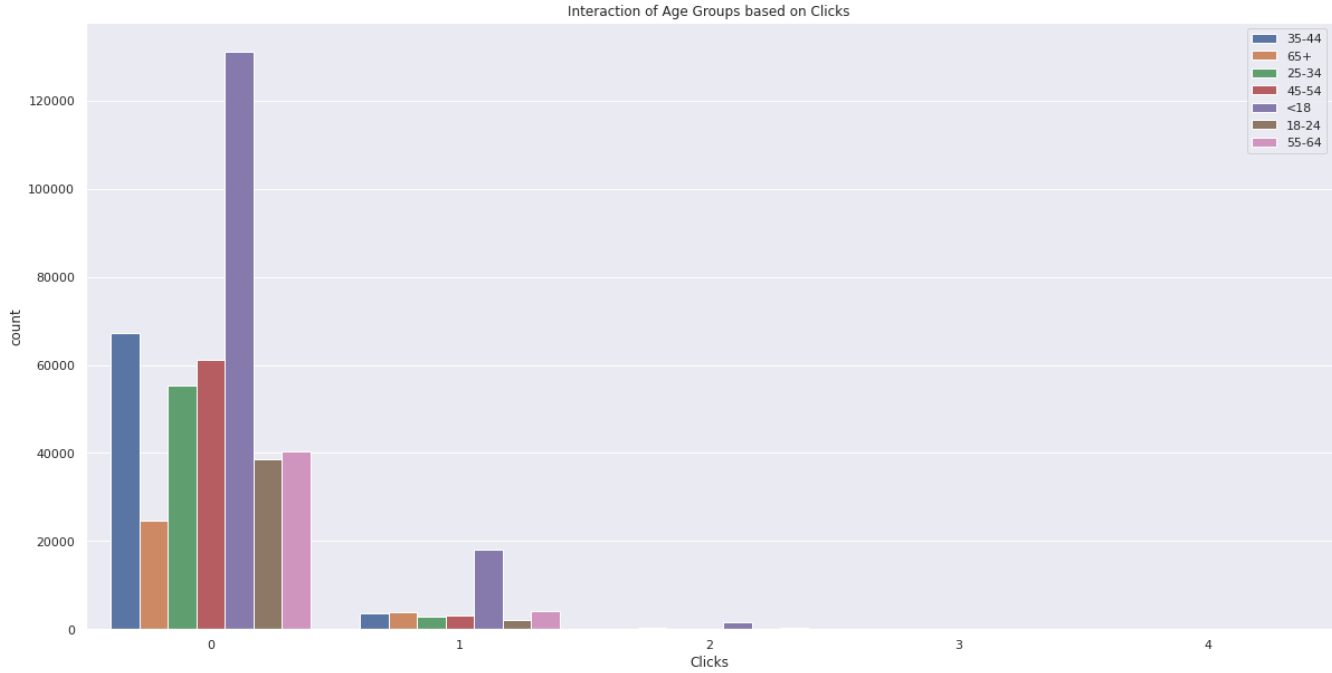


i. Define a new variable to segment or categorize users based on their click behavior

```
# interaction = []
# for row in df1b.itertuples(index=True, name='Pandas'):
#   if getattr(row, "Clicks") == 0:
#     interaction.append(0)
#   else:
#     interaction.append(1)
# df1b["Click Interaction"] = interaction
```

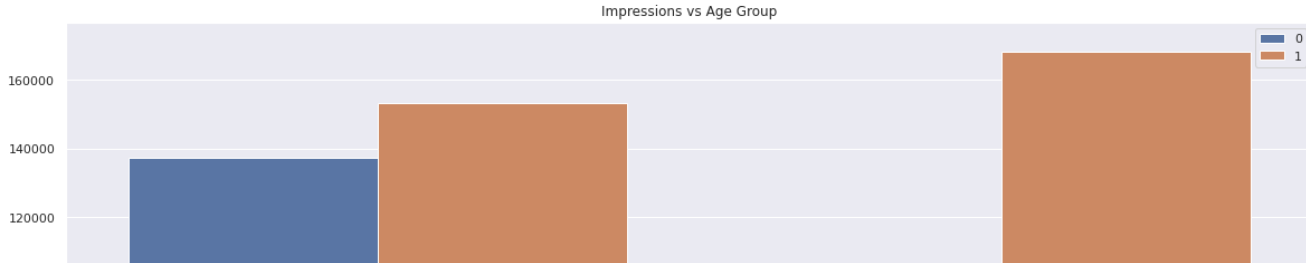
```
sns.countplot(x='Clicks', hue='Age Group', data=df1b)
plt.legend(loc='upper right')
plt.title("Interaction of Age Groups based on Clicks")
```

```
Text(0.5, 1.0, 'Interaction of Age Groups based on Clicks')
```




```
sns.countplot(x='Gender', hue='Signed_In', data=df1b)
plt.legend(loc='upper right')
plt.title("Impressions vs Age Group")
```

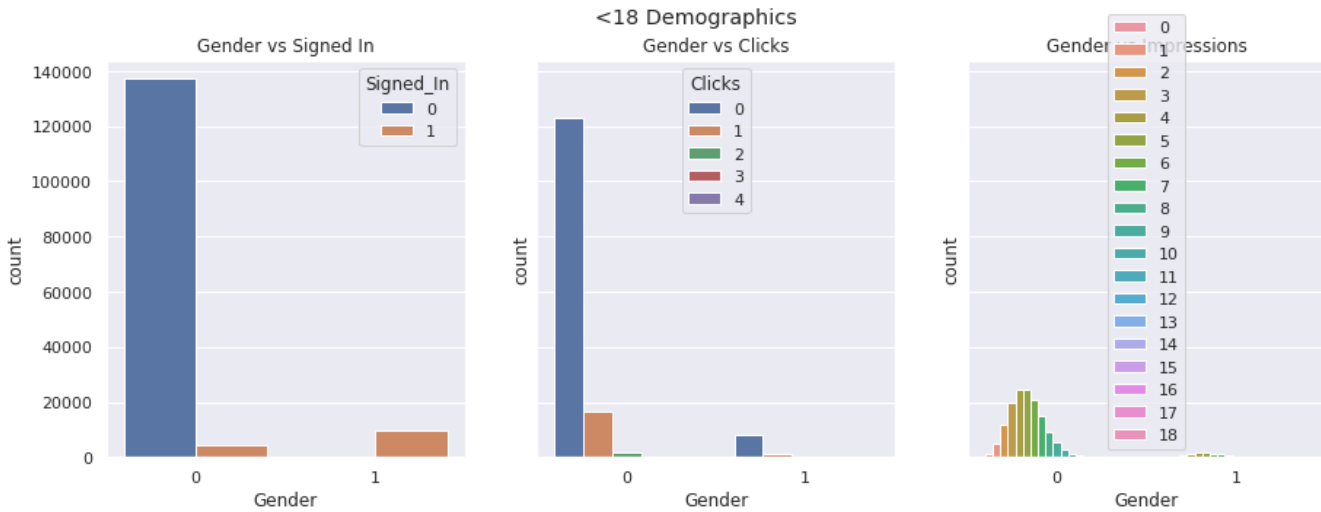
```
Text(0.5, 1.0, 'Impressions vs Age Group')
```



ii. Explore the data and make visual and quantitative comparisons across user segments/ demographics (<18 year old male vs < 18 year old females or logged-in vs not, for example).

```
fag = input("Enter the age group you wish to see the demographics for: \n 1 for <18 \n 2 for
if fag == "1":
  ag = "<18"
elif fag == "2":
  ag = "18-24"
elif fag == "3":
  ag = "25-34"
elif fag == "4":
  ag = "35-44"
elif fag == "5":
  ag = "45-54"
elif fag == "6":
  ag = "55-64"
elif fag == "7":
  ag = "65+"
else:
  print("Enter valid number")

under18 = df1b[df1b["Age Group"] == ag]

fig, axes = plt.subplots(1, 3, figsize=(15, 5), sharey=True)
fig.suptitle(ag+' Demographics')

# Gender vs Signed In
sns.countplot(ax=axes[0],data=under18, x="Gender", hue="Signed_In")
axes[0].set_title("Gender vs Signed In")

# Gender vs Clicks
sns.countplot(ax=axes[1],data=under18, x="Gender", hue="Clicks")
axes[1].set_title("Gender vs Clicks")

# Gender vs Impressions
sns.countplot(ax=axes[2],data=under18, x="Gender", hue="Impressions").legend(loc="top right")
axes[2].set_title("Gender vs Impressions")
```

```
Enter the age group you wish to see the demographics for:
 1 for <18
 2 for 18-24
 3 for 25-34
 4 for 35-44
 5 for 45-54
 6 for 55-64
 7 for 65+
1
Text(0.5, 1.0, 'Gender vs Impressions')
```



c. Create metrics/measurements/statistics that summarize the data. Examples of potential metrics include CTR, quantiles, mean, median, variance, max, and these can be calculated across the various user segments. Be selective. Think about what will be important to track over time; what will compress the data, but still capture user behavior. Now extend your analysis across days (one week is sufficient). Visualize metrics and distributions over time. Your plot should emphasize what actually changes over days

```
dfc1 = df1b
dfc2 = df[458440:(458440+449936)]
dfc3 = df[(458440+449936):(458440+449936+440371)]
dfc4 = df[(458440+449936+440371):(458440+449936+440371+442858)]
dfc5 = df[(458440+449936+440371+442858):(458440+449936+440371+442858+370329)]
dfc6 = df[(458440+449936+440371+442858+370329):(458440+449936+440371+442858+370329+764511)]
dfc7 = df[(458440+449936+440371+442858+370329+764511):(458440+449936+440371+442858+370329+764
```

```
# Gender vs Signed In
sns.countplot(ax=axes[0],data=under18, x="Gender", hue="Signed_In")
axes[0].set_title("Gender vs Signed In")

# Gender vs Clicks
sns.countplot(ax=axes[1],data=under18, x="Gender", hue="Clicks")
```

```
axes[1].set_title("Gender vs Clicks")

# Gender vs Impressions
sns.countplot(ax=axes[2],data=under18,·x="Gender",·hue="Impressions").legend(loc="top·right"
axes[2].set_title("Gender·vs·Impressions")
```

```
Text(0.5, 1.0, 'Gender vs Impressions')
```

```
dfcs = [dfc1,dfc2,dfc3,dfc4,dfc5,dfc6,dfc7]
Age = []
Gender = []
Impressions = []
Clicks = []
CTR = []
for dfc in dfcs:
  quantiles = dfc.quantile(.5, interpolation="nearest")
  mean = dfc.mean()
  median = dfc.median()
  variance = dfc.var()
  max = dfc.max()
  Age.append([quantiles[0].astype(np.float),mean[0].astype(np.float),median[0].astype(np.floa
  Gender.append([quantiles[1].astype(np.float),mean[1].astype(np.float),median[1].astype(np.f
  Impressions.append([quantiles[2].astype(np.float),mean[2].astype(np.float),median[2].astype
  Clicks.append([quantiles[3].astype(np.float),mean[3].astype(np.float),median[3].astype(np.f
  CTR.append([quantiles[4].astype(np.float),mean[1].astype(np.float),median[4].astype(np.floa
```

```
for dfc in dfcs:
  print(df.corr())
```

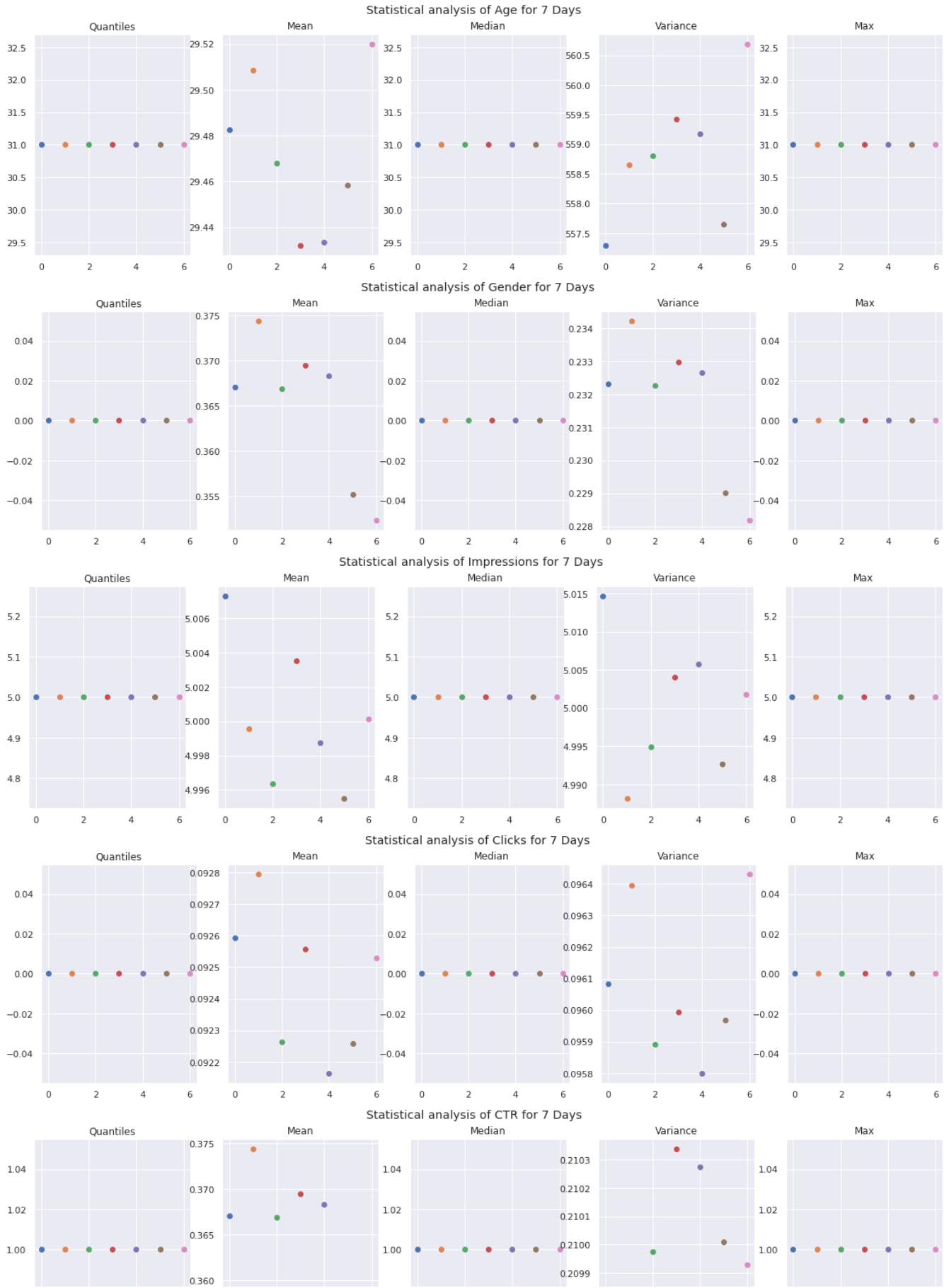
```
                  Age    Gender  Impressions    Clicks  Signed_In       CTR
Age          1.000000  0.418021    -0.000516 -0.069204   0.845325 -0.060795
Gender       0.418021  1.000000    -0.000712 -0.060767   0.536957 -0.053469
Impressions -0.000516 -0.000712     1.000000  0.137056  -0.000579  0.003273
Clicks      -0.069204 -0.060767     0.137056  1.000000  -0.107565  0.856752
Signed_In    0.845325  0.536957    -0.000579 -0.107565   1.000000 -0.094578
CTR         -0.060795 -0.053469     0.003273  0.856752  -0.094578  1.000000
                  Age    Gender  Impressions    Clicks  Signed_In       CTR
Age          1.000000  0.418021    -0.000516 -0.069204   0.845325 -0.060795
Gender       0.418021  1.000000    -0.000712 -0.060767   0.536957 -0.053469
Impressions -0.000516 -0.000712     1.000000  0.137056  -0.000579  0.003273
Clicks      -0.069204 -0.060767     0.137056  1.000000  -0.107565  0.856752
Signed_In    0.845325  0.536957    -0.000579 -0.107565   1.000000 -0.094578
CTR         -0.060795 -0.053469     0.003273  0.856752  -0.094578  1.000000
                  Age    Gender  Impressions    Clicks  Signed_In       CTR
Age          1.000000  0.418021    -0.000516 -0.069204   0.845325 -0.060795
Gender       0.418021  1.000000    -0.000712 -0.060767   0.536957 -0.053469
Impressions -0.000516 -0.000712     1.000000  0.137056  -0.000579  0.003273
Clicks      -0.069204 -0.060767     0.137056  1.000000  -0.107565  0.856752
Signed_In    0.845325  0.536957    -0.000579 -0.107565   1.000000 -0.094578
CTR         -0.060795 -0.053469     0.003273  0.856752  -0.094578  1.000000
                  Age    Gender  Impressions    Clicks  Signed_In       CTR
```

```
Age          1.000000  0.418021    -0.000516 -0.069204   0.845325 -0.060795
Gender       0.418021  1.000000    -0.000712 -0.060767   0.536957 -0.053469
Impressions -0.000516 -0.000712     1.000000  0.137056  -0.000579  0.003273
Clicks      -0.069204 -0.060767     0.137056  1.000000  -0.107565  0.856752
Signed_In    0.845325  0.536957    -0.000579 -0.107565   1.000000 -0.094578
CTR         -0.060795 -0.053469     0.003273  0.856752  -0.094578  1.000000
                  Age    Gender  Impressions    Clicks  Signed_In       CTR
Age          1.000000  0.418021    -0.000516 -0.069204   0.845325 -0.060795
Gender       0.418021  1.000000    -0.000712 -0.060767   0.536957 -0.053469
Impressions -0.000516 -0.000712     1.000000  0.137056  -0.000579  0.003273
Clicks      -0.069204 -0.060767     0.137056  1.000000  -0.107565  0.856752
Signed_In    0.845325  0.536957    -0.000579 -0.107565   1.000000 -0.094578
CTR         -0.060795 -0.053469     0.003273  0.856752  -0.094578  1.000000
                  Age    Gender  Impressions    Clicks  Signed_In       CTR
Age          1.000000  0.418021    -0.000516 -0.069204   0.845325 -0.060795
Gender       0.418021  1.000000    -0.000712 -0.060767   0.536957 -0.053469
Impressions -0.000516 -0.000712     1.000000  0.137056  -0.000579  0.003273
Clicks      -0.069204 -0.060767     0.137056  1.000000  -0.107565  0.856752
Signed_In    0.845325  0.536957    -0.000579 -0.107565   1.000000 -0.094578
CTR         -0.060795 -0.053469     0.003273  0.856752  -0.094578  1.000000
                  Age    Gender  Impressions    Clicks  Signed_In       CTR
Age          1.000000  0.418021    -0.000516 -0.069204   0.845325 -0.060795
Gender       0.418021  1.000000    -0.000712 -0.060767   0.536957 -0.053469
Impressions -0.000516 -0.000712     1.000000  0.137056  -0.000579  0.003273
Clicks      -0.069204 -0.060767     0.137056  1.000000  -0.107565  0.856752
Signed_In    0.845325  0.536957    -0.000579 -0.107565   1.000000 -0.094578
CTR         -0.060795 -0.053469     0.003273  0.856752  -0.094578  1.000000
```

In this part of the problem all the metrics and their correlation to each other has been given for 7 days and it shows that the age and the signing in has the highest correlation among all the days and it also shows that with increase in age the number of people signing also increases. This a common pattern noticed below in the graphs as well. The graphs help us to visualize the change in the statistical values over the period of 7 days and as clearly visible quantiles, median and max values remain the same for all 7 days whearas other values change with each day and hence depend a lot more on the rest on the data for the rest of the metrics. A mean of 29(approx) was noticed for age and hence middle aged people seem to often sign on higher than others.

```
stats = [Age,Gender,Impressions,Clicks,CTR]
statss = ["Age","Gender","Impressions","Clicks","CTR"]
for i in range(len(stats)):
  x = stats[i]
  y = statss[i]
  fig, ax = plt.subplots(1, 5, figsize=(20, 5))
  title = "Statistical analysis of "+y+" for 7 Days"
  fig.suptitle(title)
  for i in range(0,7):
    ax[0].scatter(i,x[i][0])
    ax[0].set_title("Quantiles")
    ax[1].scatter(i,x[i][1])
```

```
    ax[1].set_title("Mean")
    ax[2].scatter(i,x[i][2])
    ax[2].set_title("Median")
    ax[3].scatter(i,x[i][3])
    ax[3].set_title("Variance")
    ax[4].scatter(i,x[i][0])
    ax[4].set_title("Max")
```



d) Describe and interpret any patterns you find.

Have added them above along with the steps.

2) Your Data (50 points) For the second part of this assignment, you need to find some data of your own. After you've found the data that you plan to use, post about it on Piazza (also explain how you found it). I want everybody to have different data, so posting about it on ELMS will make it off limits to everybody else (this is an incentive to get it done early). The data do not have to be publicly accessible (e.g. you can use personal / professional data), but you should have every right to distribute and discuss the data (don't do anything sketchy to get the data).
a.) Thoroughly describe the data using the analysis and visualization techniques we covered in class (but feel free to go beyond them). I should come away with an understanding of your data.

```
from pandas.io.parsers.readers import read_csv
df2 = read_csv("/content/drive/MyDrive/808W Assignments/HW1/Dataset2/laptop_price.csv",encodi
print(df2)
```

```
      laptop_ID Company                            Product  \
0             1   Apple                          MacBook Pro
1             2   Apple                          Macbook Air
2             3      HP                               250 G6
3             4   Apple                          MacBook Pro
4             5   Apple                          MacBook Pro
...         ...     ...                                  ...
1298       1316  Lenovo                       Yoga 500-14ISK
1299       1317  Lenovo                       Yoga 900-13ISK
1300       1318  Lenovo                    IdeaPad 100S-14IBR
1301       1319      HP  15-AC110nv (i7-6500U/6GB/1TB/Radeon
1302       1320    Asus  X553SA-XX031T (N3050/4GB/500GB/W10)

                TypeName  Inches                          ScreenResolution  \
0              Ultrabook    13.3          IPS Panel Retina Display 2560x1600
1              Ultrabook    13.3                                   1440x900
2               Notebook    15.6                          Full HD 1920x1080
3              Ultrabook    15.4          IPS Panel Retina Display 2880x1800
4              Ultrabook    13.3          IPS Panel Retina Display 2560x1600
...                  ...     ...                                        ...
1298  2 in 1 Convertible    14.0   IPS Panel Full HD / Touchscreen 1920x1080
1299  2 in 1 Convertible    13.3  IPS Panel Quad HD+ / Touchscreen 3200x1800
1300            Notebook    14.0                                   1366x768
1301            Notebook    15.6                                   1366x768
1302            Notebook    15.6                                   1366x768

                                        Cpu   Ram               Memory  \
0                     Intel Core i5 2.3GHz   8GB            128GB SSD
1                     Intel Core i5 1.8GHz   8GB  128GB Flash Storage
2               Intel Core i5 7200U 2.5GHz   8GB            256GB SSD
3                     Intel Core i7 2.7GHz  16GB            512GB SSD
4                     Intel Core i5 3.1GHz   8GB            256GB SSD
...                                    ...   ...                  ...
1298            Intel Core i7 6500U 2.5GHz   4GB            128GB SSD
1299            Intel Core i7 6500U 2.5GHz  16GB            512GB SSD
1300  Intel Celeron Dual Core N3050 1.6GHz   2GB   64GB Flash Storage
```

```
1301            Intel Core i7 6500U 2.5GHz   6GB              1TB HDD
1302  Intel Celeron Dual Core N3050 1.6GHz   4GB            500GB HDD

                                Gpu       OpSys  Weight  Price_euros
0     Intel Iris Plus Graphics 640       macOS  1.37kg      1339.69
1           Intel HD Graphics 6000       macOS  1.34kg       898.94
2            Intel HD Graphics 620       No OS  1.86kg       575.00
3               AMD Radeon Pro 455       macOS  1.83kg      2537.45
4     Intel Iris Plus Graphics 650       macOS  1.37kg      1803.60
...                             ...         ...     ...          ...
1298         Intel HD Graphics 520  Windows 10   1.8kg       638.00
1299         Intel HD Graphics 520  Windows 10   1.3kg      1499.00
1300             Intel HD Graphics  Windows 10   1.5kg       229.00
1301            AMD Radeon R5 M330  Windows 10  2.19kg       764.00
1302             Intel HD Graphics  Windows 10   2.2kg       369.00

[1303 rows x 13 columns]
```

Now clearly we can see that though the data has been collected well, the data points often are strings instead of numbers and this might result in difficulty to perform operations on them. We need to clean the data up by converting the strings to numbers to make the data usable.

```
df2.corr()
```

	laptop_ID	Inches	Price_euros
laptop_ID	1.000000	-0.087796	0.067830
Inches	-0.087796	1.000000	0.068197
Price_euros	0.067830	0.068197	1.000000

The strings in the columns are removed using the .replace() method in pandas.

```
df2["Weight"] = df2["Weight"].str.replace('[^\d.]', '').astype(float)
df2["Ram"] = df2["Ram"].str.replace('[^\d.]', '').astype(int)
print(df2)
```

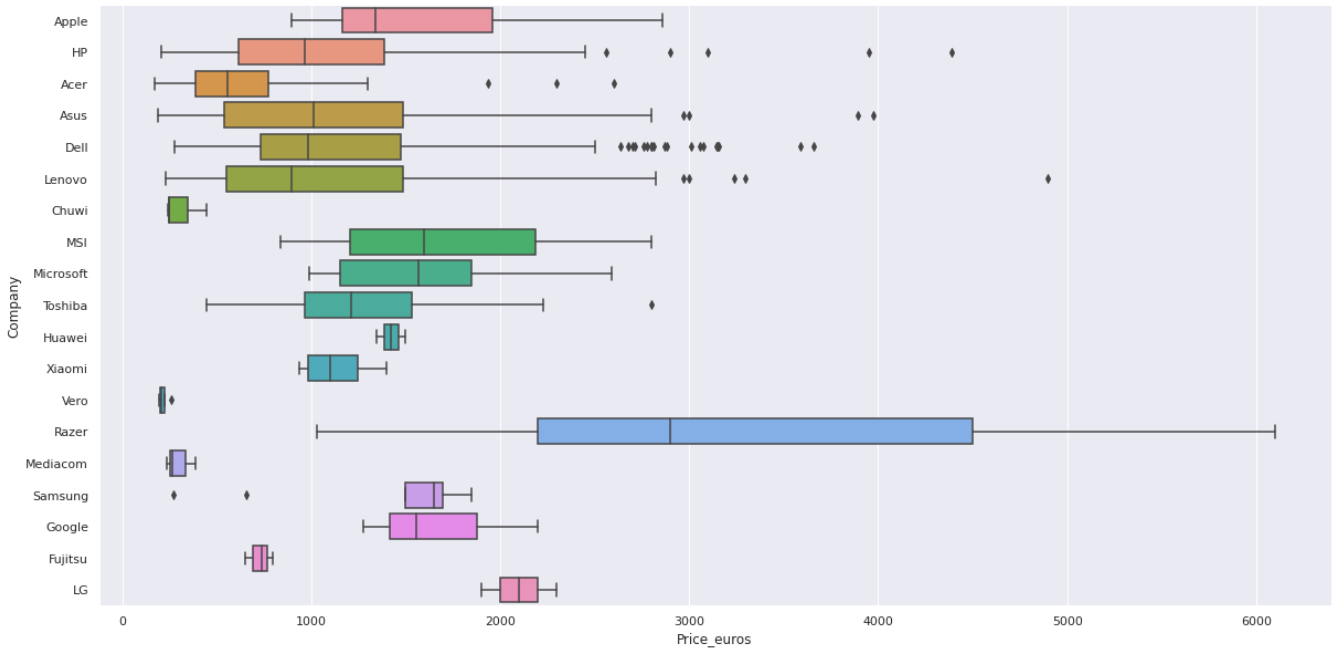
```
      laptop_ID Company                             Product  \
0             1   Apple                          MacBook Pro
1             2   Apple                          Macbook Air
2             3      HP                               250 G6
3             4   Apple                          MacBook Pro
4             5   Apple                          MacBook Pro
...         ...     ...                                  ...
1298       1316  Lenovo                       Yoga 500-14ISK
1299       1317  Lenovo                       Yoga 900-13ISK
1300       1318  Lenovo                    IdeaPad 100S-14IBR
1301       1319      HP  15-AC110nv (i7-6500U/6GB/1TB/Radeon
1302       1320    Asus  X553SA-XX031T (N3050/4GB/500GB/W10)
```

```
                TypeName  Inches                          ScreenResolution  \
0              Ultrabook    13.3          IPS Panel Retina Display 2560x1600
1              Ultrabook    13.3                                   1440x900
2               Notebook    15.6                          Full HD 1920x1080
3              Ultrabook    15.4          IPS Panel Retina Display 2880x1800
4              Ultrabook    13.3          IPS Panel Retina Display 2560x1600
...                  ...     ...                                         ...
1298  2 in 1 Convertible    14.0   IPS Panel Full HD / Touchscreen 1920x1080
1299  2 in 1 Convertible    13.3  IPS Panel Quad HD+ / Touchscreen 3200x1800
1300            Notebook    14.0                                   1366x768
1301            Notebook    15.6                                   1366x768
1302            Notebook    15.6                                   1366x768

                                         Cpu  Ram               Memory  \
0                     Intel Core i5 2.3GHz    8            128GB SSD
1                     Intel Core i5 1.8GHz    8  128GB Flash Storage
2               Intel Core i5 7200U 2.5GHz    8            256GB SSD
3                     Intel Core i7 2.7GHz   16            512GB SSD
4                     Intel Core i5 3.1GHz    8            256GB SSD
...                                     ...  ...                  ...
1298            Intel Core i7 6500U 2.5GHz    4            128GB SSD
1299            Intel Core i7 6500U 2.5GHz   16            512GB SSD
1300  Intel Celeron Dual Core N3050 1.6GHz    2   64GB Flash Storage
1301            Intel Core i7 6500U 2.5GHz    6              1TB HDD
1302  Intel Celeron Dual Core N3050 1.6GHz    4            500GB HDD

                              Gpu       OpSys  Weight  Price_euros
0     Intel Iris Plus Graphics 640       macOS    1.37      1339.69
1           Intel HD Graphics 6000       macOS    1.34       898.94
2            Intel HD Graphics 620       No OS    1.86       575.00
3               AMD Radeon Pro 455       macOS    1.83      2537.45
4     Intel Iris Plus Graphics 650       macOS    1.37      1803.60
...                             ...         ...     ...          ...
1298         Intel HD Graphics 520  Windows 10    1.80       638.00
1299         Intel HD Graphics 520  Windows 10    1.30      1499.00
1300             Intel HD Graphics  Windows 10    1.50       229.00
1301            AMD Radeon R5 M330  Windows 10    2.19       764.00
1302             Intel HD Graphics  Windows 10    2.20       369.00

[1303 rows x 13 columns]
```

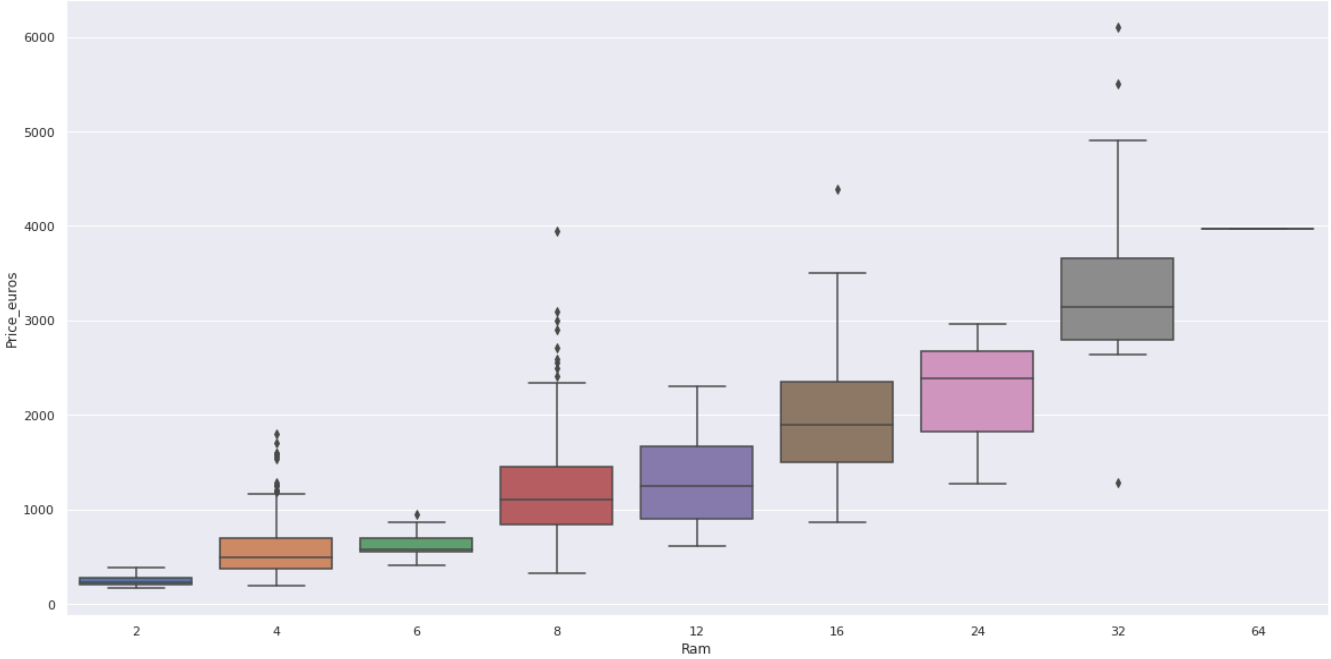
```
df2.corr()
```

	laptop_ID	Inches	Ram	Weight	Price_euros
laptop_ID	1.000000	-0.087796	-0.028607	-0.011798	0.067830
Inches	-0.087796	1.000000	0.237993	0.827631	0.068197
Ram	-0.028607	0.237993	1.000000	0.383874	0.743007
Weight	-0.011798	0.827631	0.383874	1.000000	0.210370
Price_euros	0.067830	0.068197	0.743007	0.210370	1.000000

```
sns.boxplot(y=df2["Company"], x=df2["Price_euros"])
plt.show()
```

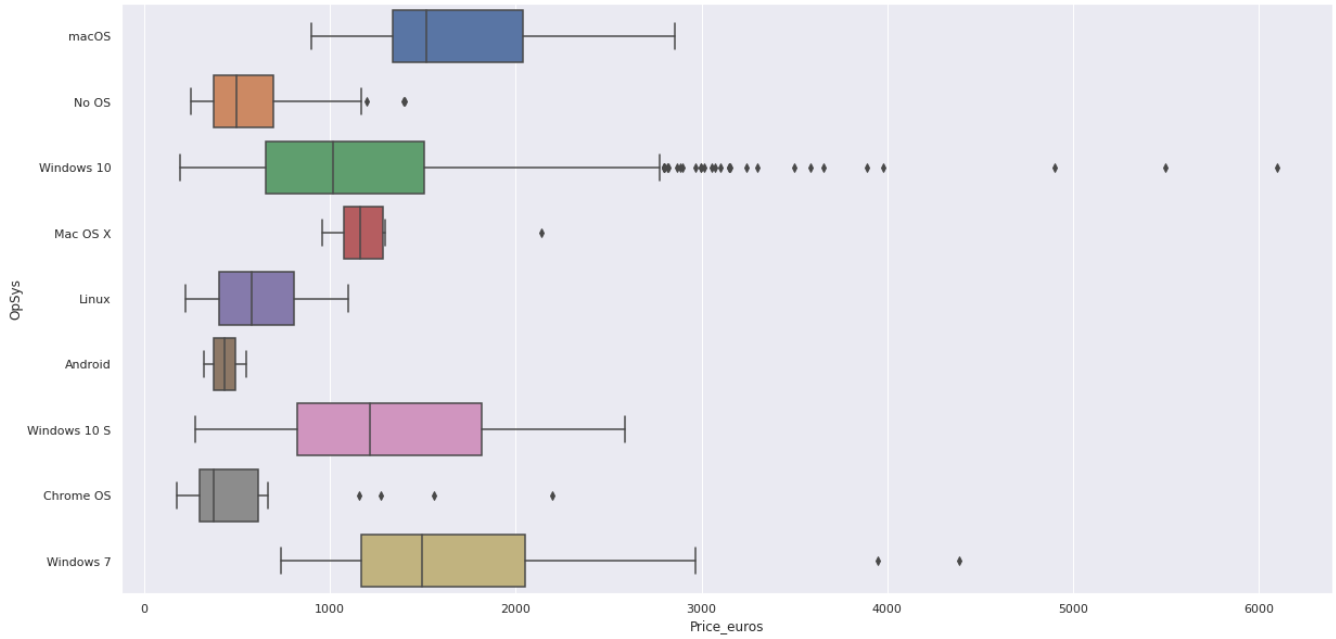



```
sns.boxplot(y=df2["Price_euros"], x=df2["Ram"])
plt.show()
```

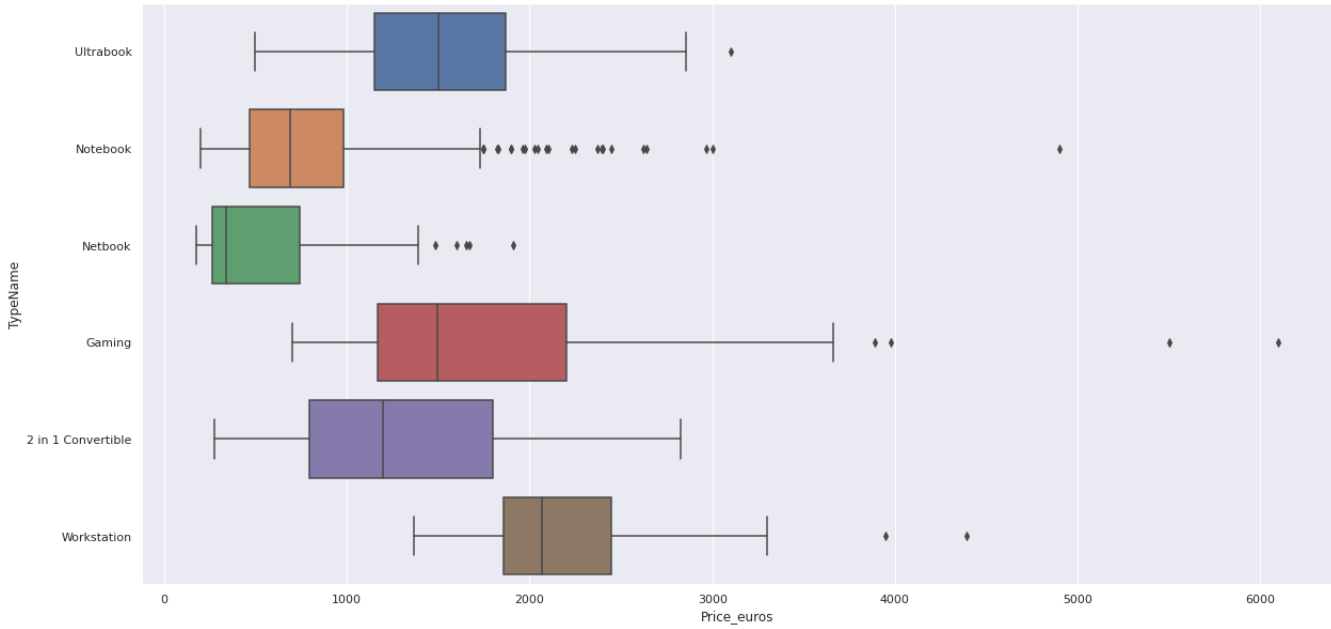



```
sns.boxplot(y=df2["Price_euros"], x=df2["Inches"])
plt.show()
```


```
sns.boxplot(y=df2["OpSys"], x=df2["Price_euros"])
plt.show()
```

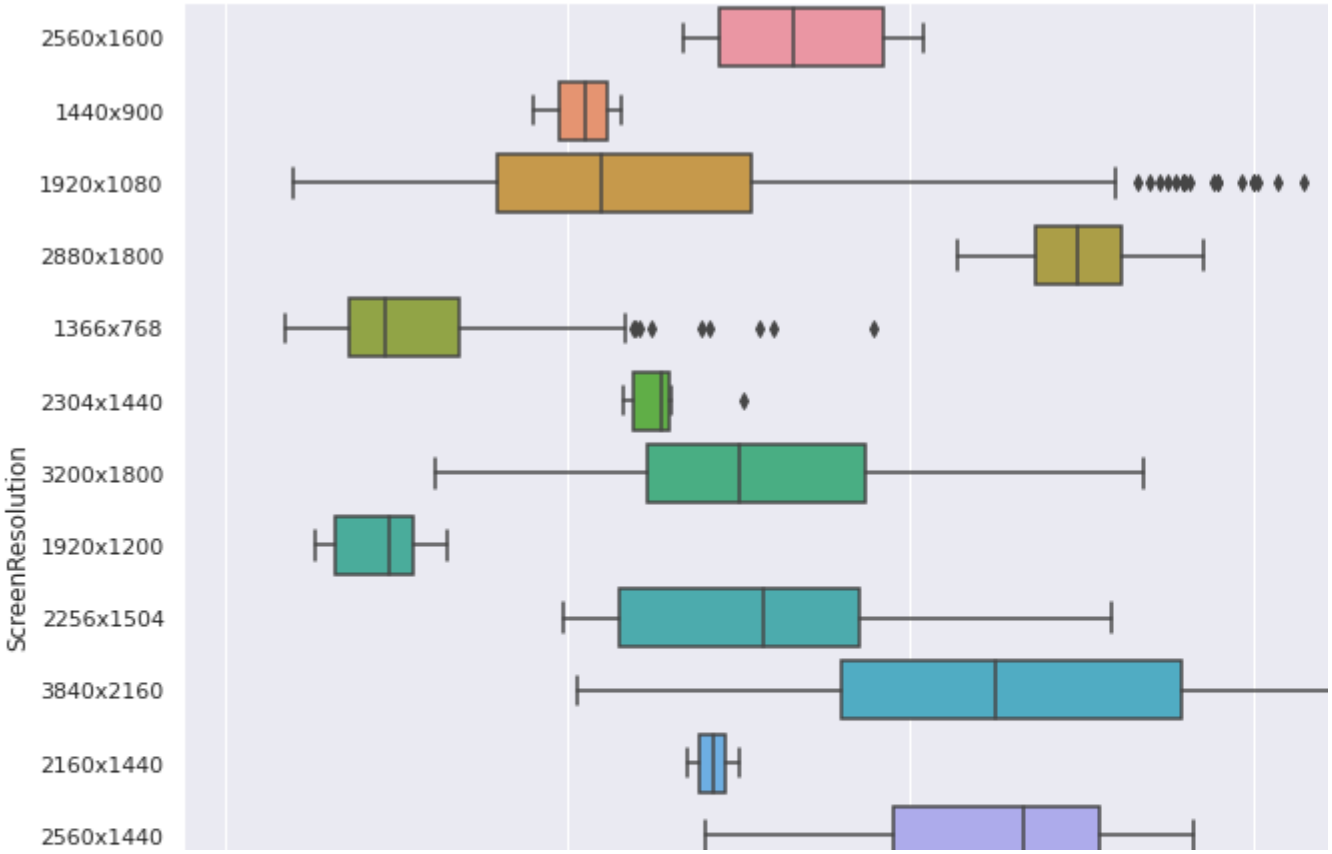



```
sns.boxplot(y=df2["TypeName"], x=df2["Price_euros"])
plt.show()
```



Since all resolutions contain 4 numbers by 3 or 4 we can remove all string characters till we obtain only the numbers and an 'x' is then added to them via a for loop.

```
df2["ScreenResolution"] = df2["ScreenResolution"].astype('string')
df2["ScreenResolution"] = df2["ScreenResolution"].str.replace('4K', '', regex=True)
df2["ScreenResolution"] = df2["ScreenResolution"].str.replace('\D', '', regex=True)
def insert_dash(string, index):
    return string[:index] + 'x' + string[index:]

i = 0
for i in range(0,len(df2["ScreenResolution"])):
    df2["ScreenResolution"][i] = insert_dash(df2["ScreenResolution"][i],4)
    i = i +1
sns.boxplot(x = df2["Price_euros"], y = df2["ScreenResolution"])
plt.show()
```



Double-click (or enter) to edit