Dataset: Netflix Userbase.csv

Context: Entertainment & Streaming

The goal of this project is to support **product and marketing teams** of streaming platforms by uncovering patterns in user behavior.
I loaded, cleaned, transformed, visualized, and exported a dataset containing user interactions across streaming services.
The analysis identifies actionable insights to improve **personalized recommendations**, **content scheduling**, and **marketing campaigns**.

Skills demonstrated: Data cleaning, transformation, visualization, storytelling
Tools: Python (Pandas, Matplotlib, Seaborn)

Step 1 - Import and Exploration

1.1 - Considering that the objective of this project is to analyze the media consumption habits of users for the elaboration of marketing campaigns, the most significant fields are:

- Subscription Type
- Device
- Country
- Acts
- Join Date

Through the analysis of these fields we can understand the behavior of a certain age group, the preference for a certain type of plan, which period there is more adherence and also verify the relationship of the number of users by each country.

1.2 - Main points observed about the categorical and numerical data:

- "Age": through the "describe()" function in this categorical variable we can observe that most of the clients are in the adult age group, that is, 25-50 years old;
- There is not a large price variation between the Standard, Premium and Basic plans;
- The Dataset is based on a duration plan only, that is, monthly;
- The categorical variables "Subscription Type", "Gender", and "Device" have a limited number of unique values, which indicates well-defined categories.
- The "Country" column has more categories, that is, more countries, which brings greater diversity to the dataset.

```
import pandas the pd

df= pd.read_csv("Netflix Userbase.csv") #---------------------------------------------------------------- print("Is the Da
```

```
O DataFrame está vazio? False
Dimensões do DataFrame: (2500, 10)

Nomes das colunas:
['User ID', 'Subscription Type', 'Monthly Revenue', 'Join Date', 'Last Payment Date', 'Country', 'Age', 'Gender', 'Device', 'Plan Duration']

Tipos de dados de cada coluna:
User ID              int64
Subscription Type    object
Monthly Revenue       int64
Join Date            object
Last Payment Date    object
Country              object
Age                   int64
Gender               object
Device               object
Plan Duration        object
dtype: object
```

```
print("First lines of the DataFrame:")
display(df.head()) print("\nLast lines of the DataFrame:")
display(df.tail()) print("\nDataFrame General Information:")
df.info() print("\nDescriptive statistics (numerical variables):")
display(df.describe()) #------------------------------------------------------------------- print("\nNumber of non-null
```

```
Primeiras linhas do DataFrame:
```

	User ID int64	Subscription Type	Monthly Revenue i.	Join Date object	Last Payment Date	Country object	Acts int64	G
0	1	Basic	10	15-01-22	10-06-23	United States	28	M
1	2	Premium	15	05-09-21	22-06-23	Canada	35	F
2	3	Standard	12	28-02-23	27-06-23	United Kingdom	42	M
3	4	Standard	12	10-07-22	26-06-23	Australia	51	F
4	5	Basic	10	01-05-23	28-06-23	Germany	33	M

5 rows, 10 cols 10 / page Page 1 of 1

```
Últimas linhas do DataFrame:
```

	User ID int64	Subscription Type	Monthly Revenue i.	Join Date object	Last Payment Date	Country object	Acts int64	G
24...	2496	Premium	14	25-07-22	12-07-23	Spain	28	F
24...	2497	Basic	15	04-08-22	14-07-23	Spain	33	F
24...	2498	Standard	12	09-08-22	15-07-23	United States	38	M
24...	2499	Standard	13	12-08-22	12-07-23	Canada	48	F
24...	2500	Basic	15	13-08-22	12-07-23	United States	35	F

5 rows, 10 cols 10 / page Page 1 of 1

```
Informações gerais do DataFrame:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2500 entries, 0 to 2499
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   User ID            2500 non-null   int64
 1   Subscription Type  2500 non-null   object
 2   Monthly Revenue    2500 non-null   int64
 3   Join Date          2500 non-null   object
 4   Last Payment Date  2500 non-null   object
 5   Country            2500 non-null   object
 6   Age                2500 non-null   int64
 7   Gender             2500 non-null   object
 8   Device             2500 non-null   object
 9   Plan Duration      2500 non-null   object
dtypes: int64(3), object(7)
memory usage: 195.4+ KB

Estatísticas descritivas (variáveis numéricas):
```

	User ID float64	Monthly Revenue f.	Acts float64
Co...	2500	2500	2500
me...	1250.5	12.5084	38.7956
Std	721.8321596	1.686851394	7.171777632
Min	1	10	26
25%	625.75	11	32
50%	1250.5	12	39
75%	1875.25	14	45
max	2500	15	51

8 rows, 3 cols 10 / page Page 1 of 1

```
User ID            1250.5000
Monthly Revenue      12.5084
Age                  38.7956
dtype: float64

Soma das variáveis numéricas:
User ID            3126250
Monthly Revenue      31271
Age                  96989
dtype: int64

Desvio padrão das variáveis numéricas:
User ID            721.832160
Monthly Revenue      1.686851
Age                  7.171778
dtype: float64

Mediana das variáveis numéricas:
User ID            1250.5
Monthly Revenue      12.0
Age                  39.0
dtype: float64
```

Step 2 - Structuring and persistence

2.1 - The necessary conversions are:

- The "Subscription Type", "Country", "Gender" and "Device" columns from the "object" type to the "category" type - the purpose of this conversion is to achieve faster analysis and lower memory consumption.
- The "Join Date" and "Last Payment Date" columns from the "object" type to the "datetime" type - the purpose of this conversion is the possibility to calculate differences and get a richer analysis.

Useful columns as filters for future dashboards:

- Subscription Type: useful for analyzing revenue, age profile, and behavior by subscription type;
- Country: Useful for a region/market analysis. It allows us to understand how the culture/habits of a country interfere in the consumption of this service;
- Gender: allows you to analyze if there is any plan preference by gender;
- Device: allows an analysis of which type of device is most used and can be used for marketing/product campaigns;
- Age: facilitates a classification by age group, which can lead to an analysis of consumer profiles;

2.3 Possible motivation for this simulation:
Propose a product pricing/valuation strategy based on the consumer's profile. Customers who use these devices more are "on the go" and may want to pay an extra price for the versatility of these devices and may still want to pay a more complete subscription (Premium) for getting a better resolution on the Smartphone/Tablet for example.

2.4 This type of structure allows for more punctual and faster analyses. In this case, we can check the behavior of customers from a certain country and with a certain subscription faster and more visually.

```python
df["Subscription Type"]= df["Subscription Type"].astype("category")
df["Country"]= df["Country"].astype("category")
df["Gender"]= df["Gender"].astype("category")
df["Device"]= df["Device"].astype("category")

df["Join Date"] = pd.to_datetime(df["Join Date"], format="%d-%m-%y", errors="coerce")
df["Last Payment Date"] = pd.to_datetime(df["Last Payment Date"], format="%d-%m-%y", errors="coerce") #-----------------

mais_antiga = df["Last Payment Date"]. min() print("First recorded payment date:", mais_antiga)

ultimo_pagamento= pd.to_datetime("2023-07-15")
limite_6_meses= ultimo_pagamento - pd. DateOffset(months=6) print("Only considering payments from:", limite_6_meses.date(

usuarios_ativos= df[
 (df["Last Payment Date"] >= limite_6_meses) &
 (df["Last Payment Date"] <= mais_recente)
] print("New DataFrame Dimensions:", usuarios_ativos.shape)

usuarios_ativos.to_csv("usuarios_ativos.csv", sep=";", encoding="utf-8", index=False)

display(usuarios_ativos.head()) #-------------------------------------------------------------------------------------
dispositivos_moveis= ["Smartphone", "Tablet"]
df_modificado["Monthly Revenue"]= df_modificado["Monthly Revenue"].astype(float)
df_modificado.loc[df_modificado["Device"].isin(dispositivos_moveis), "Monthly Revenue"] *= 1.10 print("After 10% increase
df_modificado.loc[df_modificado["Device"].isin(dispositivos_moveis), "Monthly Revenue"] /= 1.10 print("\nAfter reverting
df_multi= df_multi.set_index(["Country", "Subscription Type"])
df_multi= df_multi.sort_index() print("\nMulti-indexed DataFrame:") print(df_multi.head())

example= df_multi.loc[("Brazil", "Basic")] print(example.head())
```

```
Última data de pagamento registrada 2023-07-15 00:00:00
Primeira data de pagamento registrada: 2023-06-10 00:00:00
Considerando apenas pagamentos a partir de: 2023-01-15
Dimensões do novo DataFrame: (2500, 10)
```

	User ID int64	Subscription Type (	Monthly Revenue i.	Join Date dateti...	Last Payment Date	Country Category	Acts int64	G
0	1	Basic	10	2022-01-15 00:0...	2023-06-10 00:0...	United States	28	M
1	2	Premium	15	2021-09-05 00:0...	2023-06-22 00:0...	Canada	35	F
2	3	Standard	12	2023-02-28 00:0...	2023-06-27 00:0...	United Kingdom	42	M
3	4	Standard	12	2022-07-10 00:0...	2023-06-26 00:0...	Australia	51	F
4	5	Basic	10	2023-05-01 00:0...	2023-06-28 00:0...	Germany	33	M

5 rows, 10 cols 25 / page Page 1 of 1

```
                          Plan Duration
Country   Subscription Type
Australia Basic                 1 Month
          Basic                 1 Month
          Basic                 1 Month
          Basic                 1 Month
          Basic                 1 Month
                         User ID  Monthly Revenue  Join Date  \
Country Subscription Type
Brazil  Basic                  17               10 2022-01-24
        Basic                  27               10 2022-08-29
        Basic                  37               10 2022-09-14
        Basic                  47               10 2022-10-05
        Basic                  57               10 2022-10-01

                        Last Payment Date  Age  Gender    Device  \
Country Subscription Type
Brazil  Basic                  2023-06-25   30  Female    Laptop
        Basic                  2023-06-25   47  Female  Smart TV
        Basic                  2023-06-25   45  Female    Laptop
        Basic                  2023-06-25   39  Female  Smart TV
        Basic                  2023-06-25   46  Female    Laptop

                        Plan Duration
Country Subscription Type
Brazil  Basic                 1 Month
        Basic                 1 Month
        Basic                 1 Month
        Basic                 1 Month
        Basic                 1 Month
```

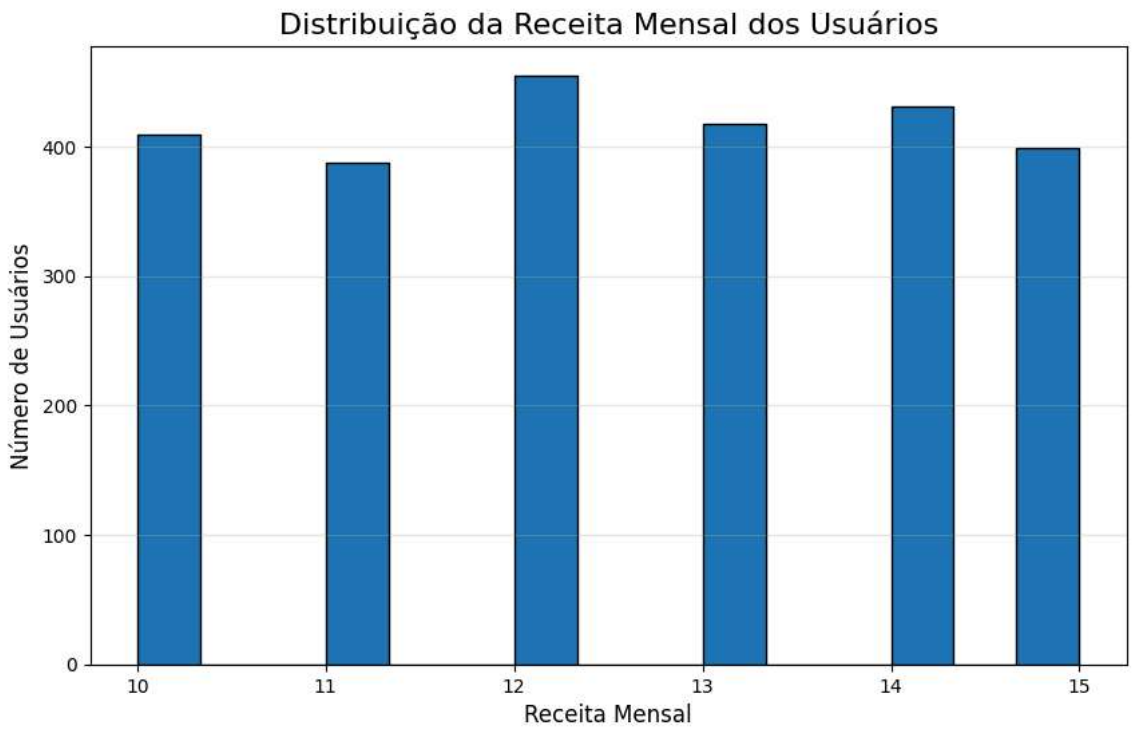
Step 3 - Exploratory visualization

3.1 By viewing this figure, we can see that there are different values for the same subscription plan; that there are no outliers, that is, very high and/or very low amounts of monthly revenue; The lowest revenue amount is 10 and the highest is 15.

3.2 The median is concentrated in customers between 37 and 38 years old and does not vary much between the three plans. From what the figure indicates, there are no outliers; the minimum and maximum ages are very close, and in the Premium plan the minimum is a little lower compared to the other two plans.

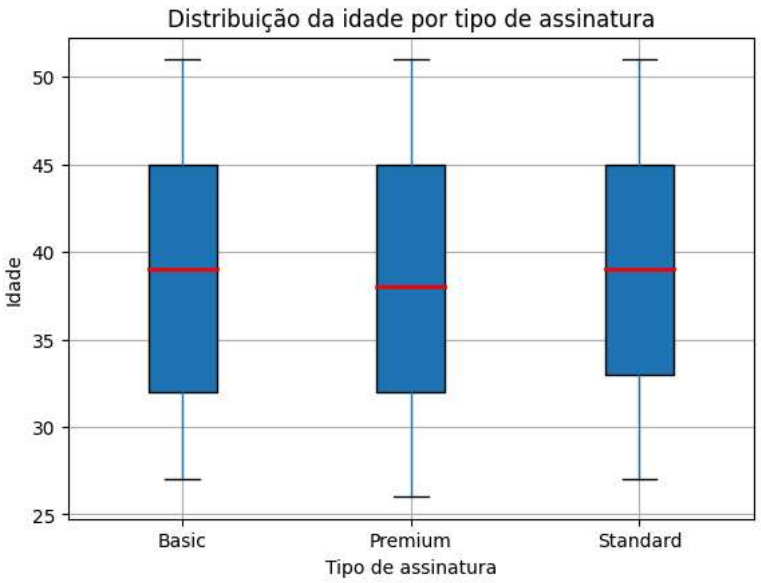
3.4 With the visualization of this scatter plot, we can observe that, in general, the same age groups are present in all monthly income groups, there are no significant groupings that indicate a disparity between the values x ages. Despite this "regularity" we can observe an outlier that indicates a group of customers aged 26 years (the youngest) grouped in the monthly revenue "15". With this visualization we cannot decipher the exact number of customers, but we can know that they exist.

```
import matplotlib.pyplot as plt

revenue= df["Monthly Revenue"]
plt.figure(figsize=(10,6))
plt.hist(recipe, bins= 15, color= "#1f77b4", edgecolor= "black")
plt.title("Monthly User Revenue Distribution", fontsize=16)
plt.xlabel("Monthly Revenue", fontsize=12)
plt.ylabel("Number of Users", fontsize=12)

plt.grid(axis='y', alpha=0.3)
plt.show()
```

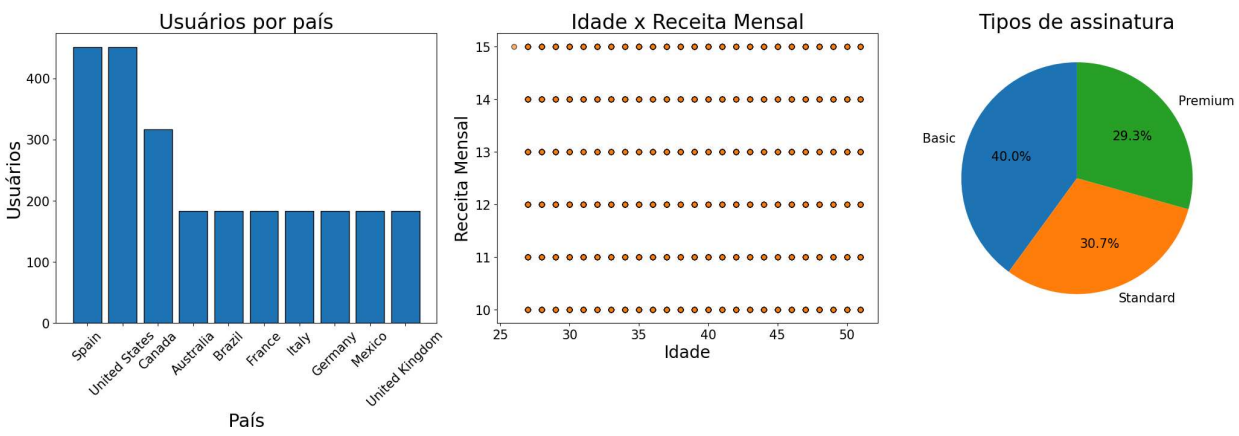


```
plt.figure(figsize=(10, 6))
df.boxplot(column="Age", by="Subscription Type", grid=True,
 patch_artist=True,
 boxprops=dict(facecolor="#1f77b4", color="black"), medianprops=dict(color="red", linewidth=2))
plt.title("Age Distribution by Subscription Type")
plt.suptitle("")
plt.xlabel("Signature Type")
plt.ylabel("Age")
plt.show()
```

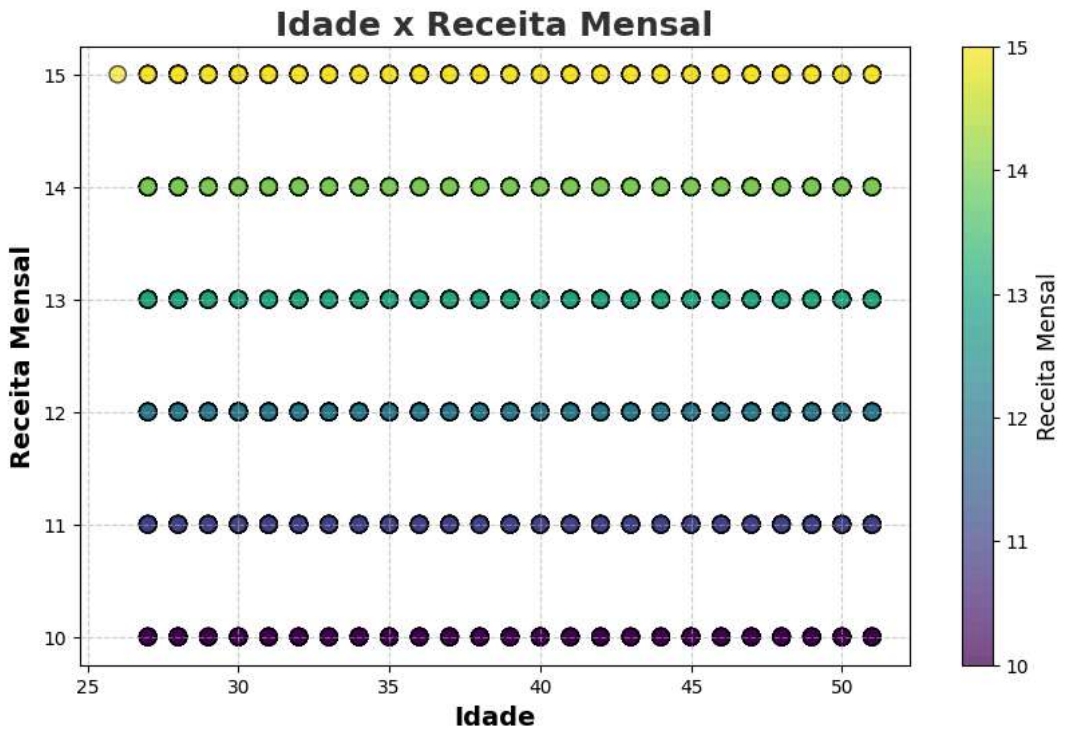
```
<Figure size 1000x600 with 0 Axes>
```




```
fig, axs= plt.subplots(1, 3, figsize=(21, 7), constrained_layout=True)

usuarios_por_pais= df["Country"].value_counts()
axs[0].bar(usuarios_por_pais.index, usuarios_por_pais.values, color="#1f77b4", edgecolor="black")
axs[0].set_title("Users by country", fontsize=24)
axs[0].set_xlabel("Country", fontsize=22)
axs[0].set_ylabel("Users", fontsize=22)
axs[0].tick_params(axis='x', labelsize=15, rotation=45)
axs[0].tick_params(axis="y", labelsize=15)

axs[1].scatter(df["Age"], df["Monthly Revenue"], alpha=0.6, c="#ff7f0e", edgecolors="black")
axs[1].set_title("Age x Monthly Income", fontsize=24)
axs[1].set_xlabel("Age", fontsize=20)
axs[1].set_ylabel("Monthly Revenue", fontsize=20)
axs[1].tick_params(axis='x', labelsize=15)
axs[1].tick_params(axis='y', labelsize=15)

subscriptions= df["Subscription Type"].value_counts()
axs[2].pie(signatures, labels=signatures.index, autopct="%1.1f%%", startangle=90, textprops={'fontsize':15}, colors=["#1f
axs[2].set_title("Signature types", fontsize=24)

plt.show()
```



```
plt.figure(figsize=(10, 6))
plt.scatter(df["Age"], df["Monthly Revenue"],
            c=df["Monthly Revenue"],
            cmap="viridis",
            alpha=0.7,
            edgecolors="black",
            s=80)
plt.title("Idade x Receita Mensal", fontsize=18, fontweight="bold", color="#333333")
plt.xlabel("Idade", fontsize=14, fontweight="bold")
plt.ylabel("Receita Mensal", fontsize=14, fontweight="bold")

plt.grid(True, linestyle="--", alpha=0.6)

cbar=plt.colorbar()
cbar.set_label("Receita Mensal", fontsize=12)

plt.show()
```



```
print(df["Plan Duration"].unique())
print(df["Plan Duration"].value_counts())

df["Plan Duration"]= df["Plan Duration"].astype(str)
df["Plan Duration"]= df["Plan Duration"].str.extract("(\d)").astype(int)

print(df["Plan Duration"].head())
print(df["Plan Duration"].dtypes)

agrupado= df.groupby(["Country", "Subscription Type"])["Plan Duration"].agg(["mean", "median"])

print(agrupado.head(20))
```

```
['1 Month']
Plan Duration
1 Month    2500
Name: count, dtype: int64
0    1
1    1
2    1
3    1
4    1
Name: Plan Duration, dtype: int64
int64
                          mean  median
Country   Subscription Type
Australia Basic            1.0     1.0
          Premium          1.0     1.0
          Standard         1.0     1.0
Brazil    Basic            1.0     1.0
          Premium          1.0     1.0
          Standard         1.0     1.0
Canada    Basic            1.0     1.0
          Premium          1.0     1.0
          Standard         1.0     1.0
France    Basic            1.0     1.0
          Premium          1.0     1.0
          Standard         NaN     NaN
Germany   Basic            1.0     1.0
          Premium          1.0     1.0
          Standard         1.0     1.0
Italy     Basic            1.0     1.0
          Premium          1.0     1.0
```

```
import numpy as np
import pandas as pd

@np.vectorize
def faixa_etaria(idade):
    if idade < 18:
        return"<18"
    elif 18 <= idade <= 30:
        return "18-30"
    elif 31 <= idade <= 45:
        return "31-45"
    elif 46 <= idade <= 60:
        return "46-60"
    else:
        return "60+"

df["Faixa Etária"]= faixa_etaria(df["Age"])
agrupado= df.groupby("Faixa Etária").agg(
    Media_Receita=("Monthly Revenue", "mean"),
    Mediana_Receita=("Monthly Revenue", "median"),
    Soma_Receita= ("Monthly Revenue", "sum"),
    Contagem_Usuarios=("User ID", "count")
)

print(agrupado)
```

```
             Media_Receita  Mediana_Receita  Soma_Receita  Contagem_Usuarios
Faixa Etária
18-30            12.638298             13.0          5346                423
31-45            12.503351             12.0         18655               1492
46-60            12.427350             12.0          7270                585
```