WEEK 1

Importing the file

```
from google.colab import files
uploaded = files.upload()
```

```
Choose Files Indian_Kids…n_Time.csv
Indian_Kids_Screen_Time.csv(text/csv) - 499126 bytes, last modified: 21/9/2025 - 100% done
Saving Indian_Kids_Screen_Time.csv to Indian_Kids_Screen_Time (1).csv
```

Load the dataset

Verified successful loading using df.head() and df.info().

```
from google.colab import files
uploaded = files.upload()
import pandas as pd

df = pd.read_csv(list(uploaded.keys())[0])
df.head()
```

```
Choose Files Indian_Kids…n_Time.csv
Indian_Kids_Screen_Time.csv(text/csv) - 499126 bytes, last modified: 21/9/2025 - 100% done
Saving Indian_Kids_Screen_Time.csv to Indian_Kids_Screen_Time.csv
```

	Age	Gender	Avg_Daily_Screen_Time_hr	Primary_Device	Exceeded_Recommended_Limit	Educational_to_Recreational_Ratio	Health_
0	14	Male	3.99	Smartphone	True	0.42	Poor Sle
1	11	Female	4.61	Laptop	True	0.30	Po
2	18	Female	3.73	TV	True	0.32	Po
3	15	Female	1.21	Laptop	False	0.39	

Next steps: Generate code with df | New interactive sheet

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9712 entries, 0 to 9711
Data columns (total 8 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Age                             9712 non-null   int64
 1   Gender                          9712 non-null   object
 2   Avg_Daily_Screen_Time_hr        9712 non-null   float64
 3   Primary_Device                  9712 non-null   object
 4   Exceeded_Recommended_Limit      9712 non-null   bool
 5   Educational_to_Recreational_Ratio  9712 non-null   float64
 6   Health_Impacts                  6494 non-null   object
 7   Urban_or_Rural                  9712 non-null   object
dtypes: bool(1), float64(2), int64(1), object(4)
memory usage: 540.7+ KB
```

Checking null values

Used df.isnull().sum() to identify missing values in each column.

Learned that handling null values is important before applying ML models.

```
df.isnull().sum()
```

	0
Age	0
Gender	0
Avg_Daily_Screen_Time_hr	0
Primary_Device	0
Exceeded_Recommended_Limit	0
Educational_to_Recreational_Ratio	0
Health_Impacts	3218
Urban_or_Rural	0

dtype: int64

Checking duplicate rows

Used df.duplicated().sum() to find duplicate rows.

Understood that duplicates can lead to data bias and wrong model training.

```
df.duplicated().sum()
```

```
np.int64(44)
```

Correlation matrix

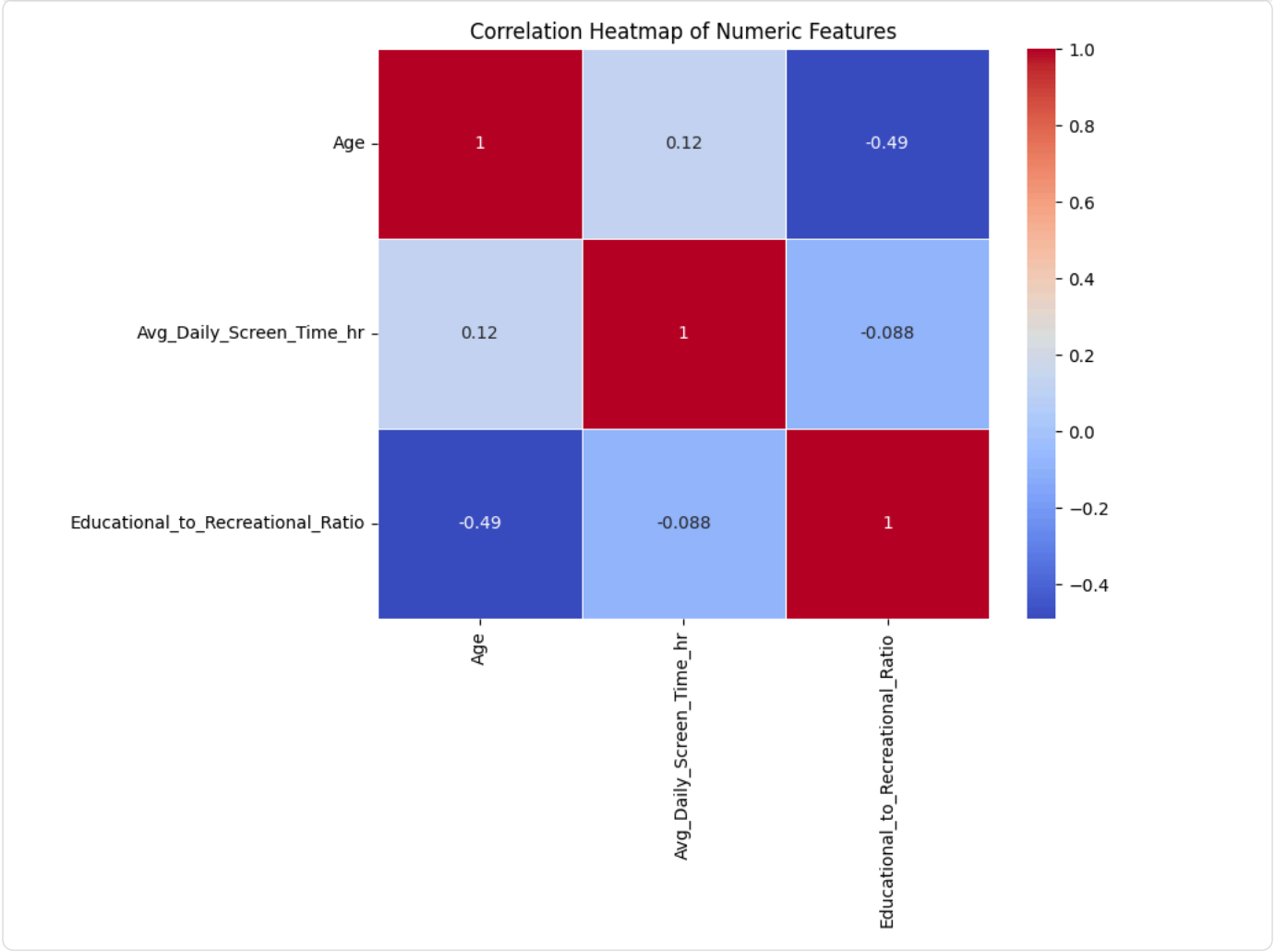
Applied df.corr() to see relationships between numerical features.

Positive correlation- one increases, the other also increases.

Negative correlation- one increases, the other decreases.

Helps in feature selection and reducing multicollinearity.

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
num_df = df.select_dtypes(include=[np.number])
corr_matrix = num_df.corr()
plt.figure(figsize=(8,6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title("Correlation Heatmap of Numeric Features")
plt.show()
```



Columns datatypes

Confirmed data types with df.dtypes.

Important step because wrong datatypes (e.g., numbers stored as strings) affect analysis.

```
pd.DataFrame(df.dtypes, columns=['dtype']).reset_index().rename(columns={'index':'column'})
```

	column	dtype
0	Age	int64
1	Gender	object
2	Avg_Daily_Screen_Time_hr	float64
3	Primary_Device	object
4	Exceeded_Recommended_Limit	bool
5	Educational_to_Recreational_Ratio	float64
6	Health_Impacts	object
7	Urban_or_Rural	object

IQR

Applied Interquartile Range (IQR) method to detect outliers.

Formula:

Q1 = 25th percentile, Q3 = 75th percentile

IQR = Q3 − Q1

Outlier Range = [Q1 − 1.5IQR, Q3 + 1.5IQR]

```
numerical_data = df.select_dtypes(include='number')
Q1 = numerical_data.quantile(0.25)
```

```
Q3 = numerical_data.quantile(0.75)
IQR = Q3 - Q1
print("Q1:\n", Q1)
print("Q3:\n", Q3)
print("IQR:\n", IQR)
```

```
Q1:
 Age                                10.00
Avg_Daily_Screen_Time_hr            3.41
Educational_to_Recreational_Ratio   0.37
Name: 0.25, dtype: float64
Q3:
 Age                                16.00
Avg_Daily_Screen_Time_hr            5.38
Educational_to_Recreational_Ratio   0.48
Name: 0.75, dtype: float64
IQR:
 Age                                6.00
Avg_Daily_Screen_Time_hr           1.97
Educational_to_Recreational_Ratio  0.11
dtype: float64
```

OBSERVATIONS

The dataset has 9712 rows and 8 columns.

The Health_Impacts column has missing values.

Other columns are complete with no missing data.

There are 44 duplicate rows in the dataset.

Outliers are detected using the IQR method in numeric columns.

Some values of Screen Time and Educational to Recreational Ratio may be outliers.

Before modeling, we need to handle missing values, remove duplicates, and manage outliers.

WEEK 2

Replace the None to Not Reported in healthimpacts

```
df['Health_Impacts'] = df['Health_Impacts'].fillna("Not Reported")
```

Create Age Bands

```
df['AgeBand'] = pd.cut(df['Age'],
                       bins=[7, 12, 16, 18],
                       labels=['Pre-Teens', 'Teenagers', 'Late-Teens'])
```

```
df.isnull().sum()
```

	0
Age	0
Gender	0
Avg_Daily_Screen_Time_hr	0
Primary_Device	0
Exceeded_Recommended_Limit	0
Educational_to_Recreational_Ratio	0
Health_Impacts	0
Urban_or_Rural	0
AgeBand	0

dtype: int64

Categorizing Health Impacts into Physical, Mental ,or Both

```
physical = ["Eye Strain","Headache","Poor Posture","Obesity","Fatigue"]
mental   = ["Poor Sleep","Anxiety","Stress","Depression","Addiction"]
def categorize_impact(impact):
    if impact == "None":
        return "No Impact"
    has_physical = any(p in impact for p in physical)
    has_mental   = any(m in impact for m in mental)

    if has_physical and has_mental:
        return "Both Physical and Mental"
    elif has_physical:
        return "Physical"
    elif has_mental:
        return "Mental"
    else:
        return "No Impact"
df["Health_Impact_Category"] = df["Health_Impacts"].apply(categorize_impact)
```

```
df.head(10)
```

	Age	Gender	Avg_Daily_Screen_Time_hr	Primary_Device	Exceeded_Recommended_Limit	Educational_to_Recreational_Ratio	Health_
0	14	Male	3.99	Smartphone	True	0.42	Poor Sle
1	11	Female	4.61	Laptop	True	0.30	Po
2	18	Female	3.73	TV	True	0.32	Po
3	15	Female	1.21	Laptop	False	0.39	Not F
4	12	Female	5.89	Smartphone	True	0.49	Poo
5	14	Female	4.88	Smartphone	True	0.44	Po
6	17	Male	2.97	TV	False	0.48	Not F
7	10	Male	2.74	TV	True	0.54	Not F
8	14	Male	4.61	Laptop	True	0.36	Poo
9	18	Male	3.24	Tablet	True	0.48	Poo Obe

Next steps: Generate code with `df` New interactive sheet

Classifying Devices into Portable or Wall-Mounted

```
def device_type(device):
    portable = ["Smartphone", "Laptop", "Tablet"]
    wall_mounted = ["TV"]

    if device in portable:
        return "Portable"
    elif device in wall_mounted:
        return "Wall-Mounted"
    else:
        return "Other"
df["Device_Category"] = df["Primary_Device"].apply(device_type)
```

```
df.head(10)
```

	Age	Gender	Avg_Daily_Screen_Time_hr	Primary_Device	Exceeded_Recommended_Limit	Educational_to_Recreational_Ratio	Health_
0	14	Male	3.99	Smartphone	True	0.42	Poor Sle
1	11	Female	4.61	Laptop	True	0.30	Po
2	18	Female	3.73	TV	True	0.32	Po
3	15	Female	1.21	Laptop	False	0.39	Not F
4	12	Female	5.89	Smartphone	True	0.49	Poo
5	14	Female	4.88	Smartphone	True	0.44	Po
6	17	Male	2.97	TV	False	0.48	Not F
7	10	Male	2.74	TV	True	0.54	Not F
8	14	Male	4.61	Laptop	True	0.36	Poo
9	18	Male	3.24	Tablet	True	0.48	Poo Obe

Next steps: Generate code with df | New interactive sheet

Categorizing Devices Based on Screen Size

```
def screen_size(device):
    if device == 'TV':
        return '>30'
    elif device in ['Smartphone', 'Laptop', 'Tablet']:
        return '<30'
    else:
        return 'Unknown'
df['Screen_Size'] = df['Primary_Device'].apply(screen_size)
```

```
df.head(10)
```

	Age	Gender	Avg_Daily_Screen_Time_hr	Primary_Device	Exceeded_Recommended_Limit	Educational_to_Recreational_Ratio	Health_
0	14	Male	3.99	Smartphone	True	0.42	Poor Sle
1	11	Female	4.61	Laptop	True	0.30	Po
2	18	Female	3.73	TV	True	0.32	Po
3	15	Female	1.21	Laptop	False	0.39	Not F
4	12	Female	5.89	Smartphone	True	0.49	Poo
5	14	Female	4.88	Smartphone	True	0.44	Po
6	17	Male	2.97	TV	False	0.48	Not F
7	10	Male	2.74	TV	True	0.54	Not F
8	14	Male	4.61	Laptop	True	0.36	Poo
9	18	Male	3.24	Tablet	True	0.48	Poo Obe

Next steps: Generate code with df | New interactive sheet

Save the cleaned dataset

```
df.to_csv("Indian_Kids_Screen_Time_cleaned.csv", index=False)
```

checking the null values

```
df.isnull().sum()
```

	0
Age	0
Gender	0
Avg_Daily_Screen_Time_hr	0
Primary_Device	0
Exceeded_Recommended_Limit	0
Educational_to_Recreational_Ratio	0
Health_Impacts	0
Urban_or_Rural	0
AgeBand	0
Health_Impact_Category	0
Device_Category	0
Screen_Size	0

dtype: int64

```
df.head()
```

	Age	Gender	Avg_Daily_Screen_Time_hr	Primary_Device	Exceeded_Recommended_Limit	Educational_to_Recreational_Ratio	Health_
0	14	Male	3.99	Smartphone	True	0.42	Poor Sl