Ninjaz

STC

الاتصالات السعودية

CAPSTONE PROJECT

PRESENTATION OUTLINE

- Introduction
- STC Dataset
- Problem statement
- Exploratory Data Analysis (EDA)
- Data cleaning
- Data pre-processing
- ML models comparison
- Future Work

stc COMPANY

Dynamism

continuously looking to improve

Devotion

desire to become a “customer centric”

Drive

looking for the best possible solutions

SAUDI VISION 2030

The Saudi vision 2030, revealed in 2016 by Crown Prince Mohammed bin Salman, is founded on three pillars: A Vibrant Society, a thriving economy, and an ambitious nation

stc AND SAUDI VISION 2030

VISION رؤية
2030
المملكة العربية السعودية
KINGDOM OF SAUDI ARABIA

Vital Society

strengthening the economy of Saudi Arabia

Thriving Economy

STC launched the Saudi Vision Cable project

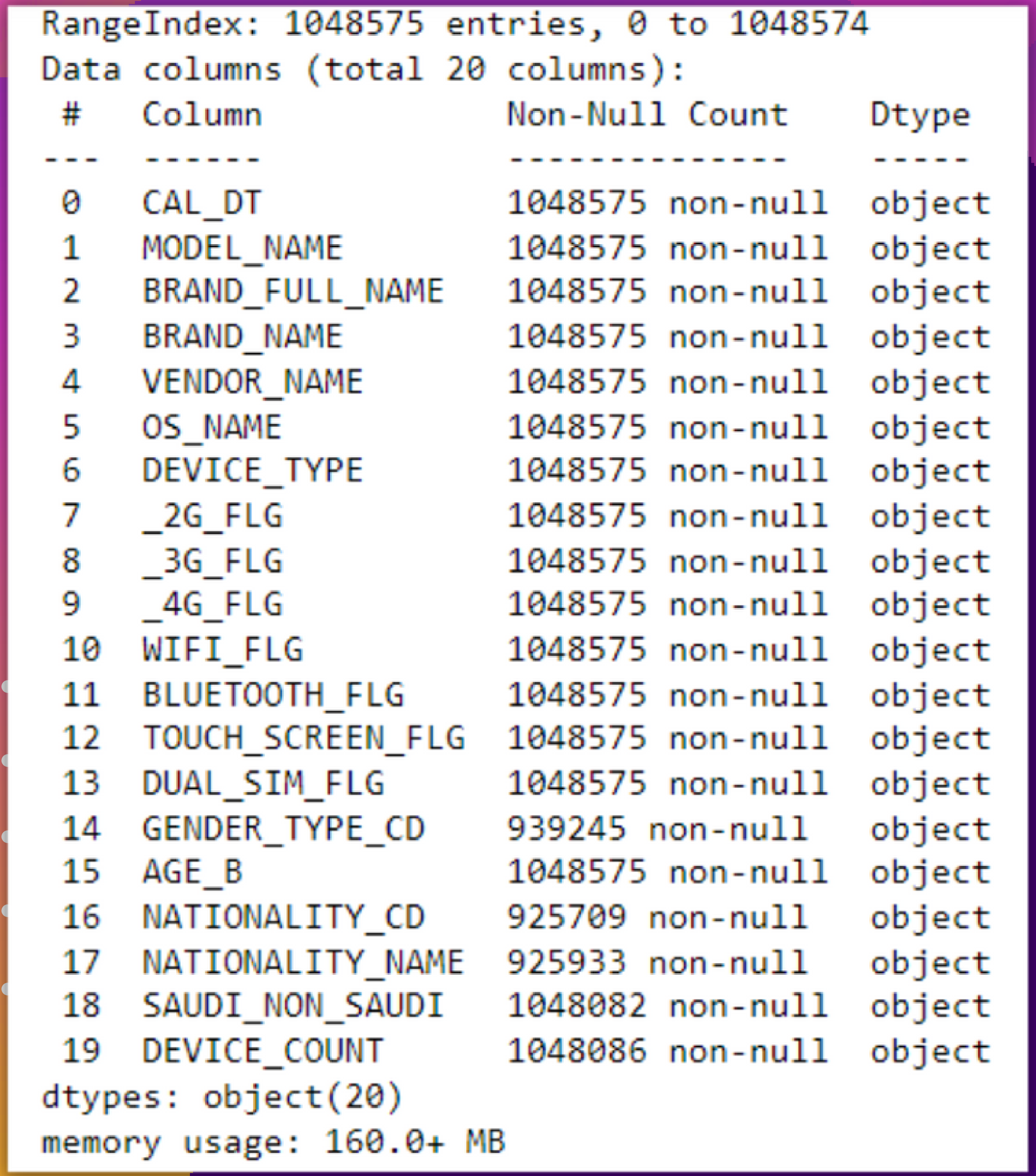
Ambitious Nation

empower the Saudis and the private companies to take better steps and continue improving

STC DATASET

The data set describes uncommon handset devices usage by customers, for an interval of 12 months and with specific customer demographics. It can be used to analyze some devices trends over time, and the devices used by different groups of customers.

The dataset contains 714023 rows, and has the following attributes:



```
RangeIndex: 1048575 entries, 0 to 1048574
Data columns (total 20 columns):
 #   Column             Non-Null Count    Dtype 
---  ------             --------------    ----- 
 0   CAL_DT             1048575 non-null  object
 1   MODEL_NAME         1048575 non-null  object
 2   BRAND_FULL_NAME    1048575 non-null  object
 3   BRAND_NAME         1048575 non-null  object
 4   VENDOR_NAME        1048575 non-null  object
 5   OS_NAME            1048575 non-null  object
 6   DEVICE_TYPE        1048575 non-null  object
 7   _2G_FLG            1048575 non-null  object
 8   _3G_FLG            1048575 non-null  object
 9   _4G_FLG            1048575 non-null  object
 10  WIFI_FLG           1048575 non-null  object
 11  BLUETOOTH_FLG      1048575 non-null  object
 12  TOUCH_SCREEN_FLG   1048575 non-null  object
 13  DUAL_SIM_FLG       1048575 non-null  object
 14  GENDER_TYPE_CD     939245 non-null   object
 15  AGE_B              1048575 non-null  object
 16  NATIONALITY_CD     925709 non-null   object
 17  NATIONALITY_NAME   925933 non-null   object
 18  SAUDI_NON_SAUDI    1048082 non-null  object
 19  DEVICE_COUNT       1048086 non-null  object
dtypes: object(20)
memory usage: 160.0+ MB
```

PROBLEM STATMENT

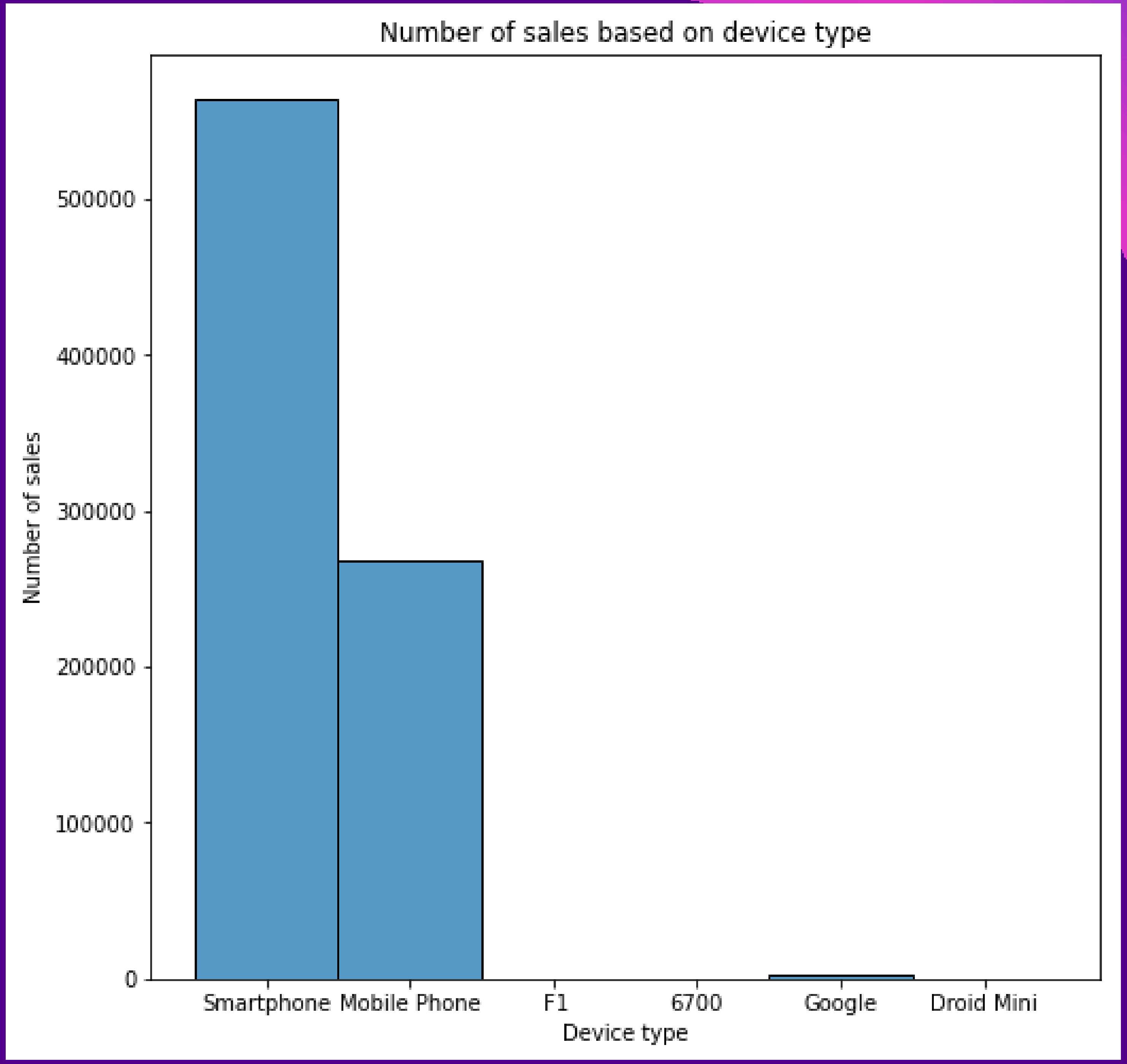
- Number of device types that are not commonly used

- stc's sales performance

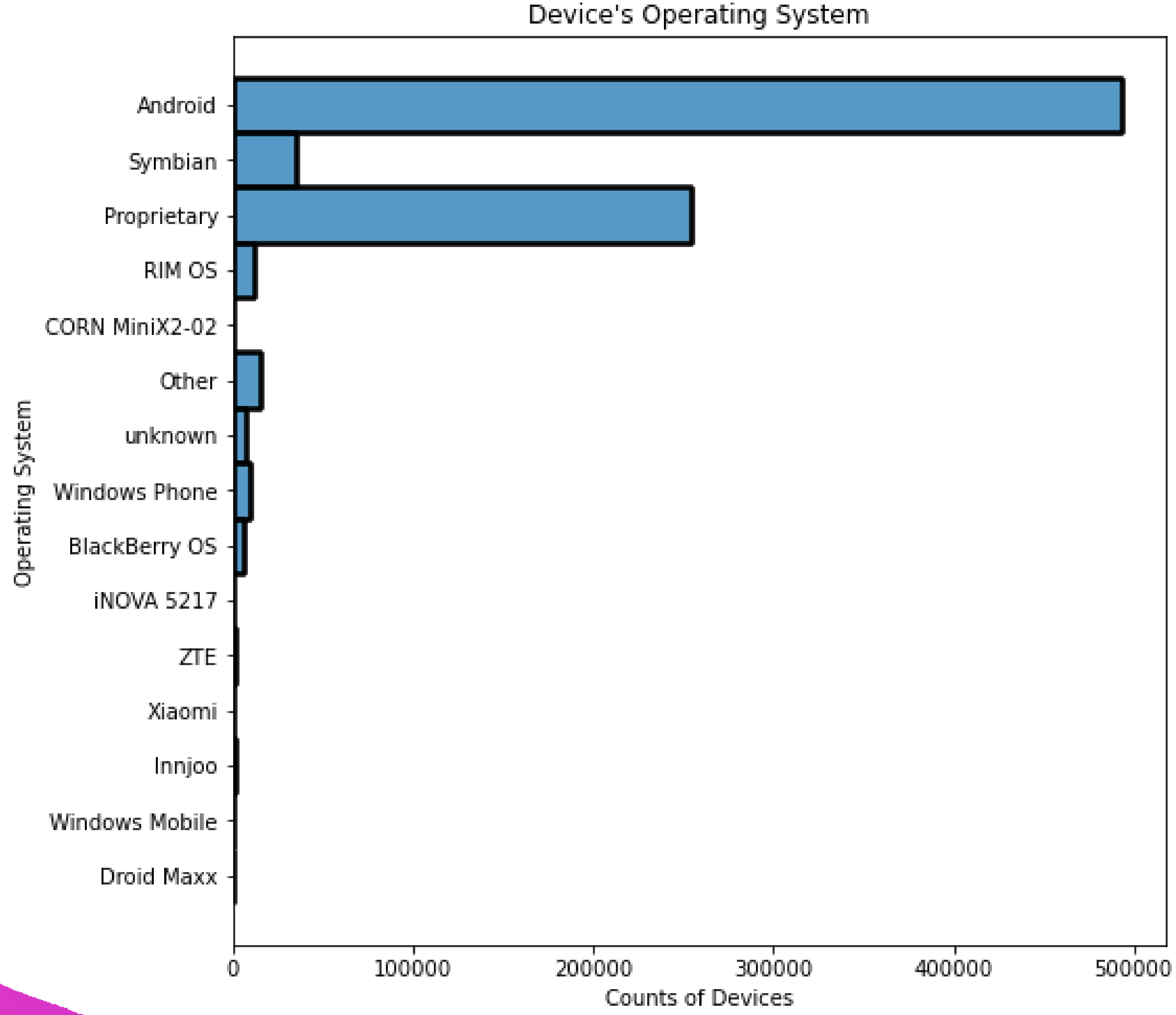
EDA

Sales

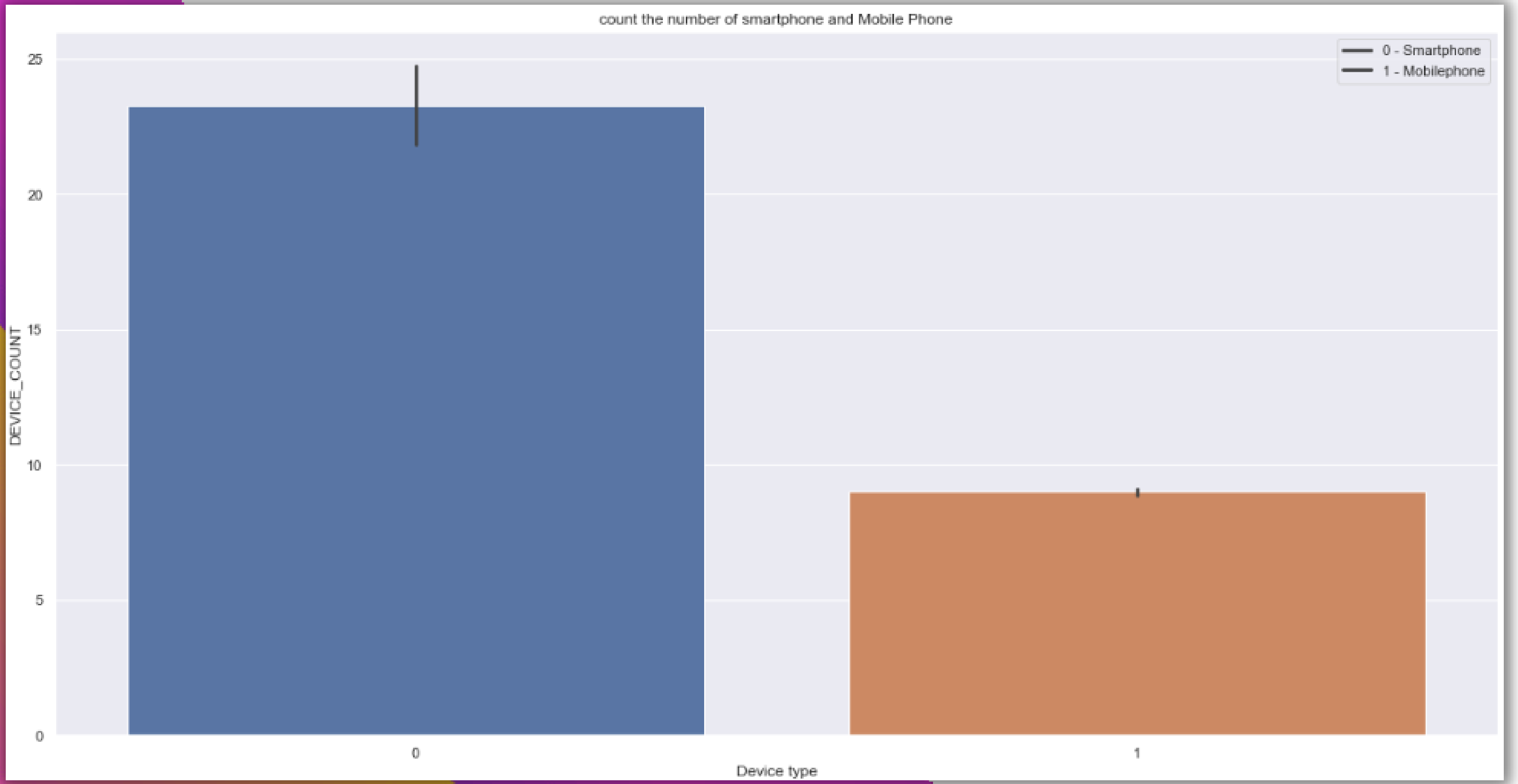
Number of sales based on device type



OPERATING SYSTEMS



This is the count of devices sold based on operating system

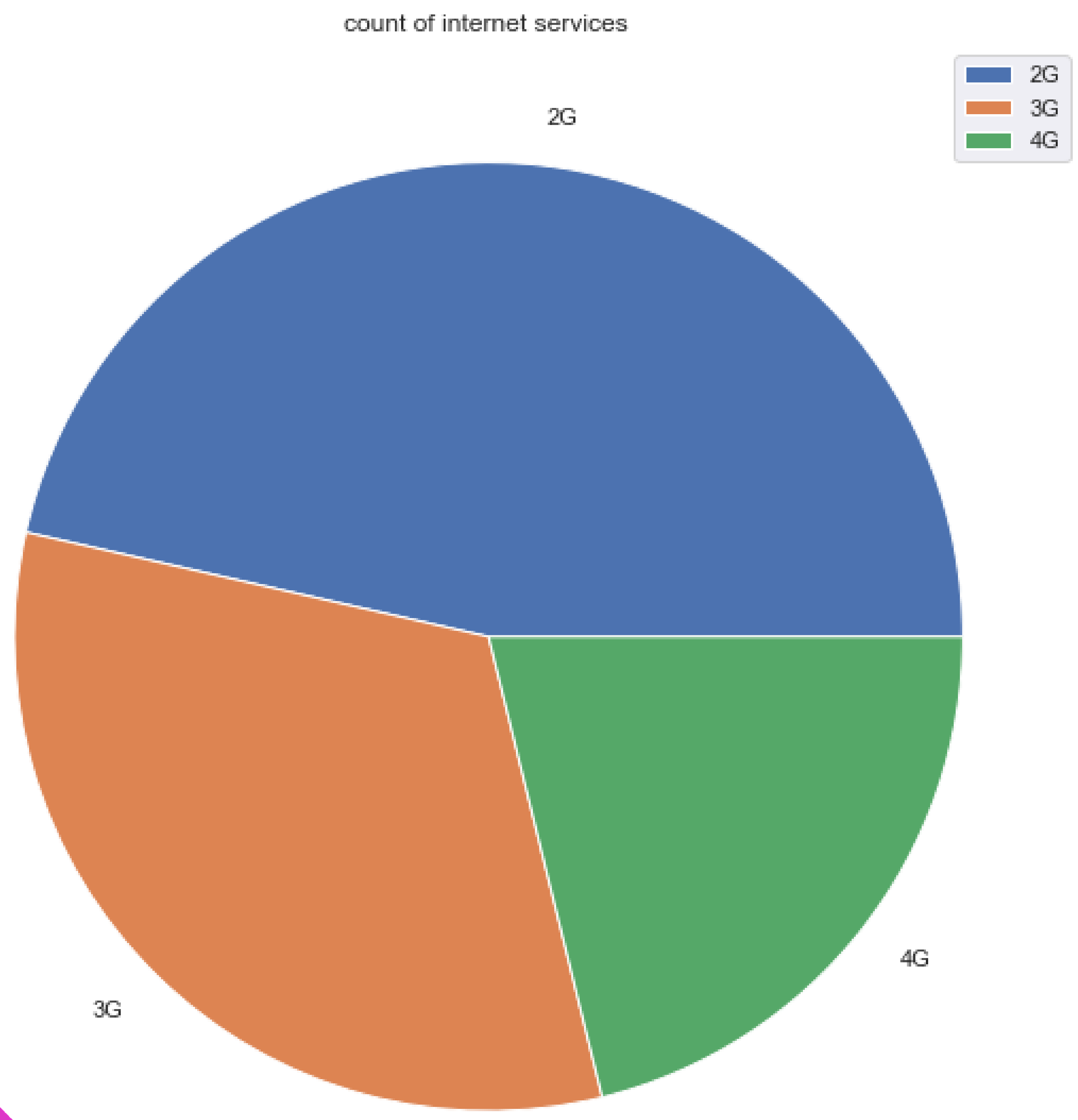


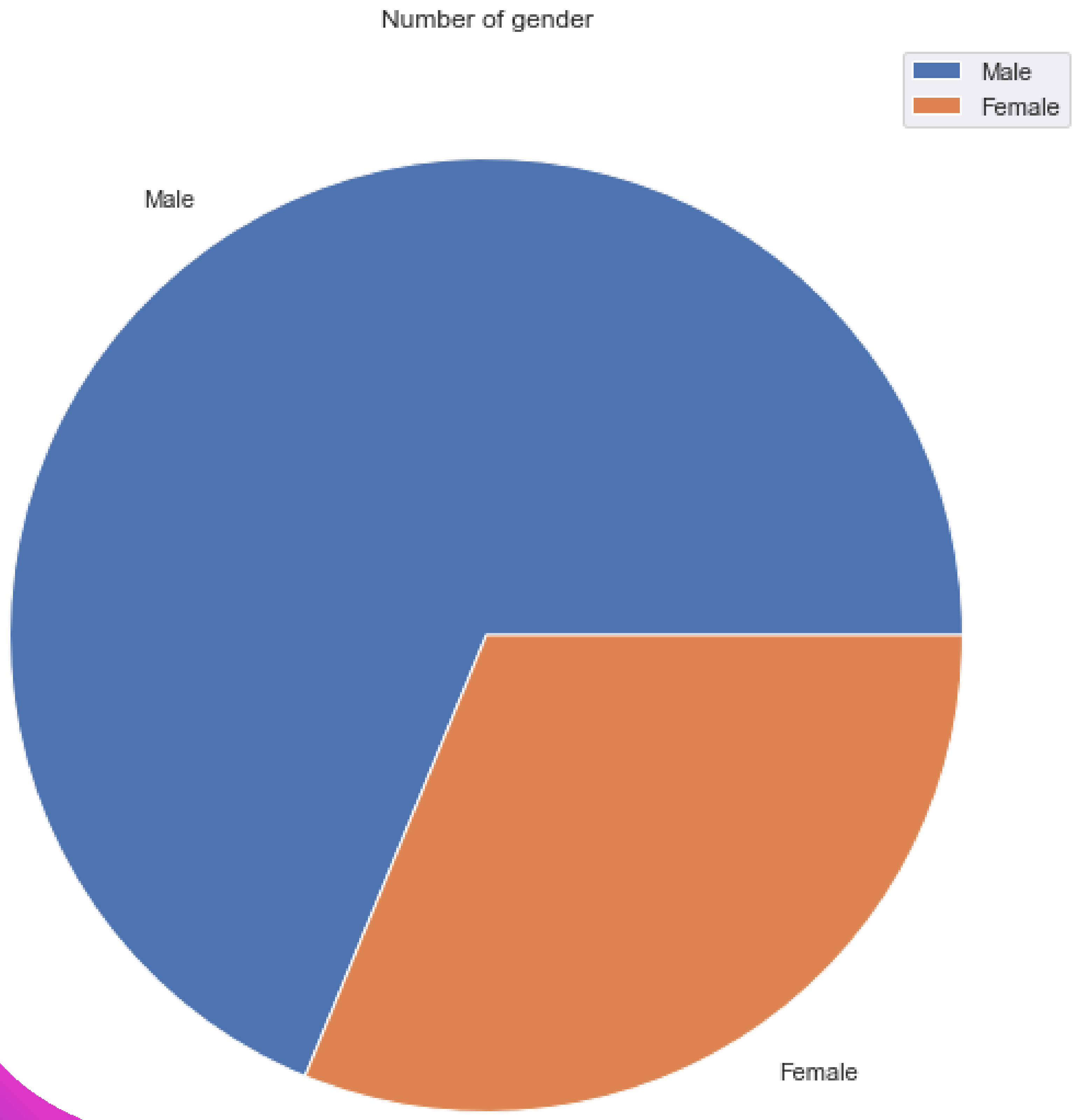
Smartphone Mobile Phone

this chart shows the count of smartphones and mobile phones devices

INTERNET SERVICES

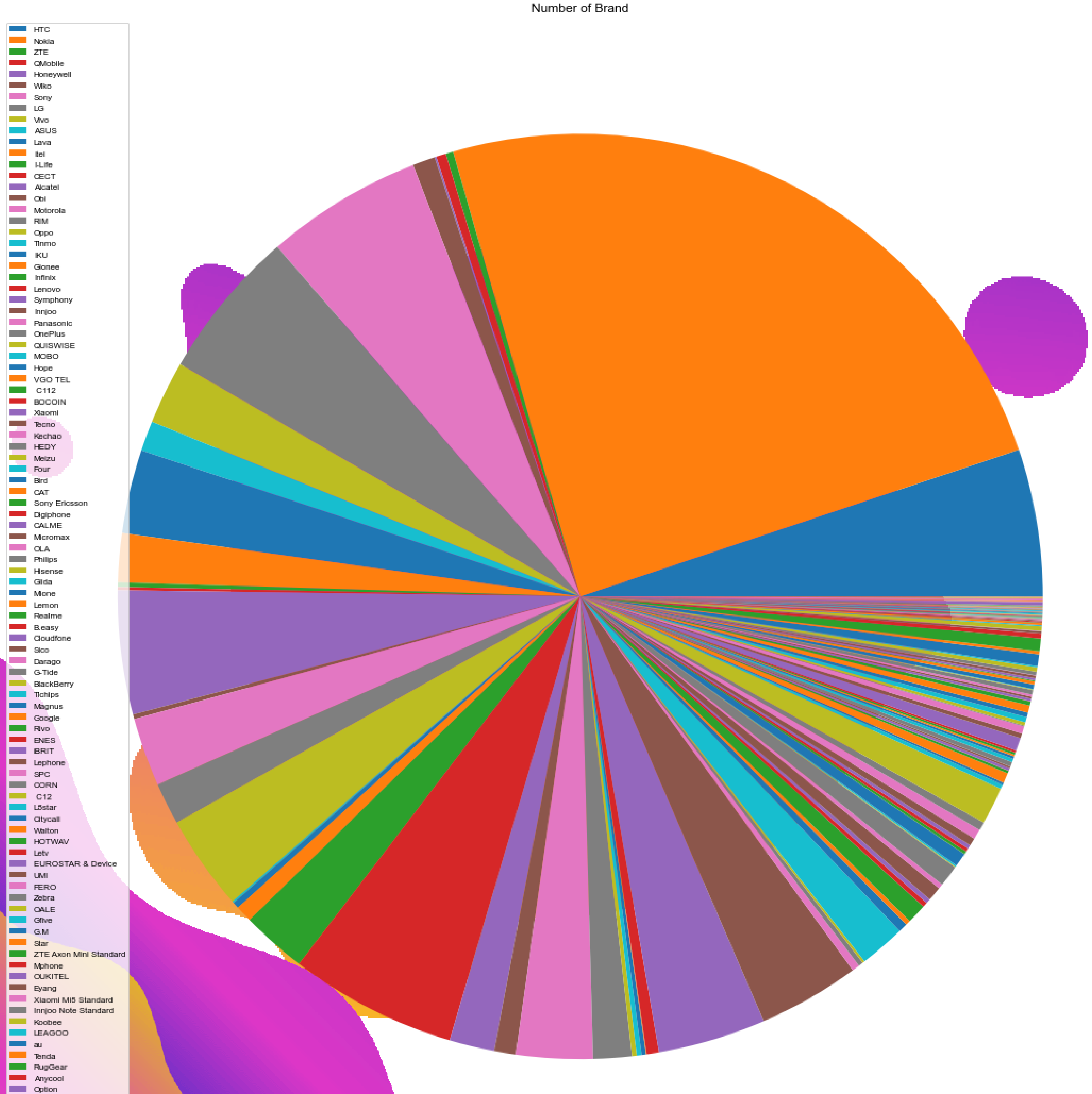
the count of devices that have 2G, 3G and 4G internet service





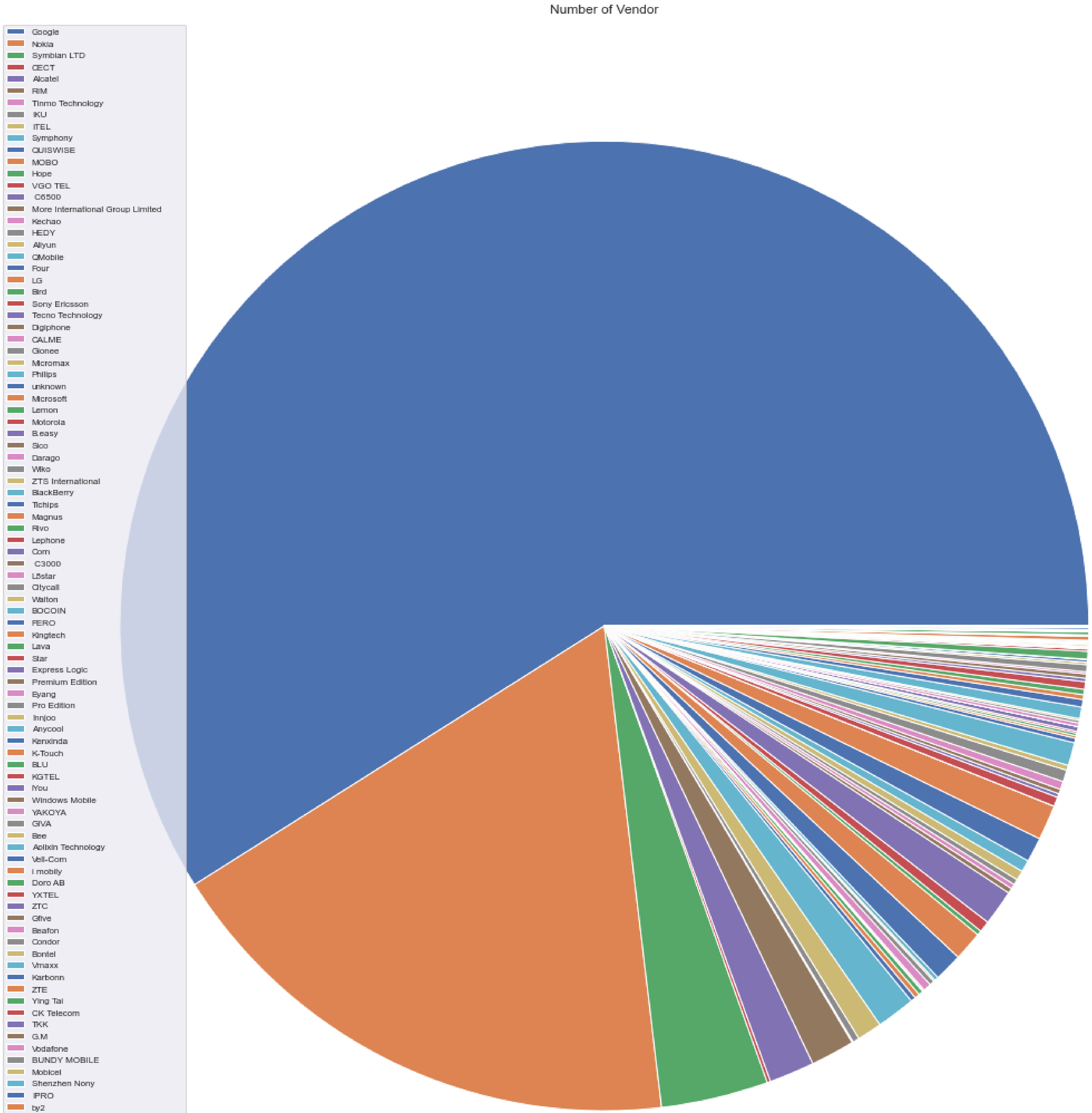
Sales Based on Gender

this chart shows number of devices sold based on gender



The number of the brands

this chart shows the number of each brand

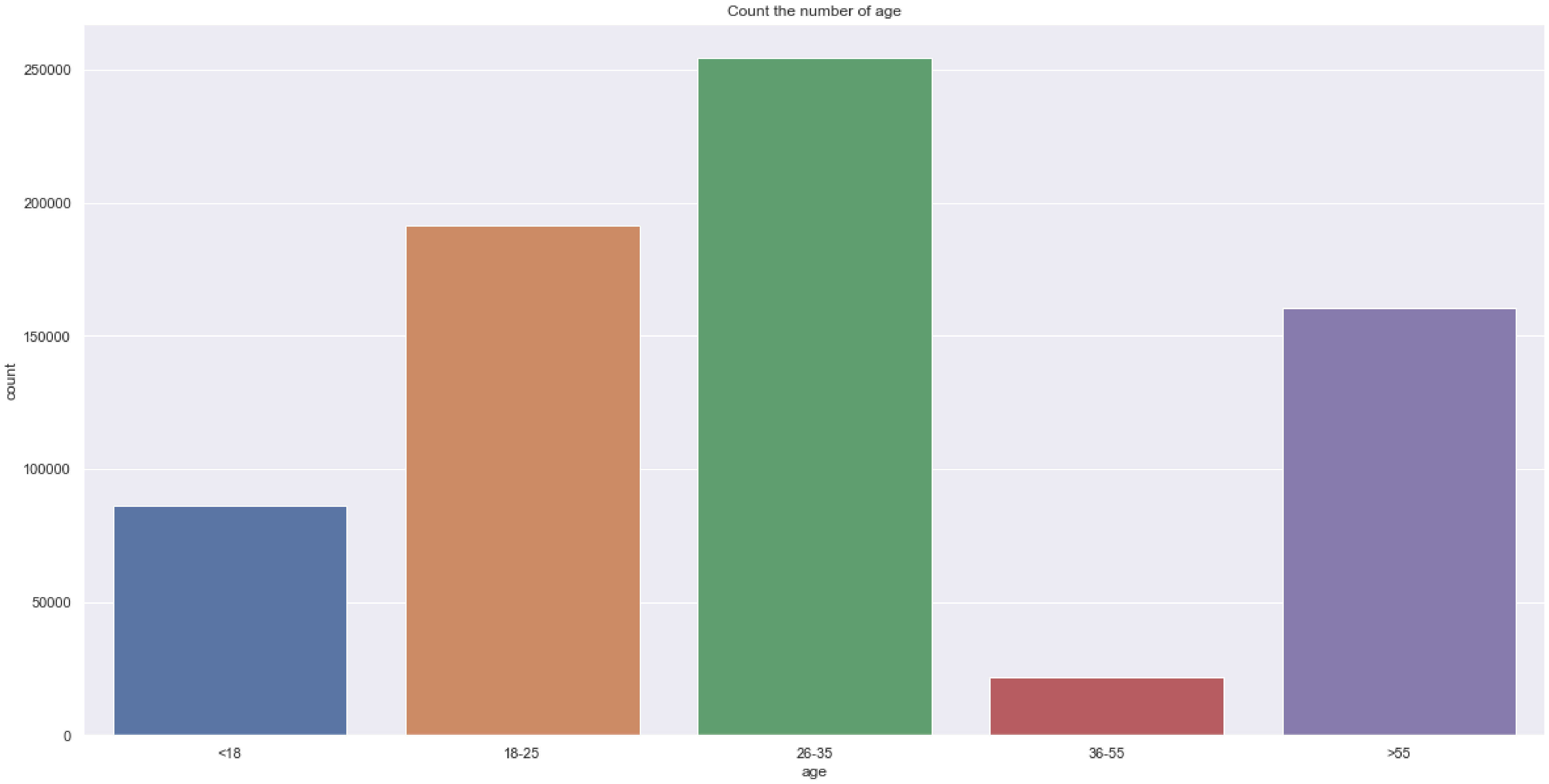


The Number of Vendors

this chart shows the number of each Vendor

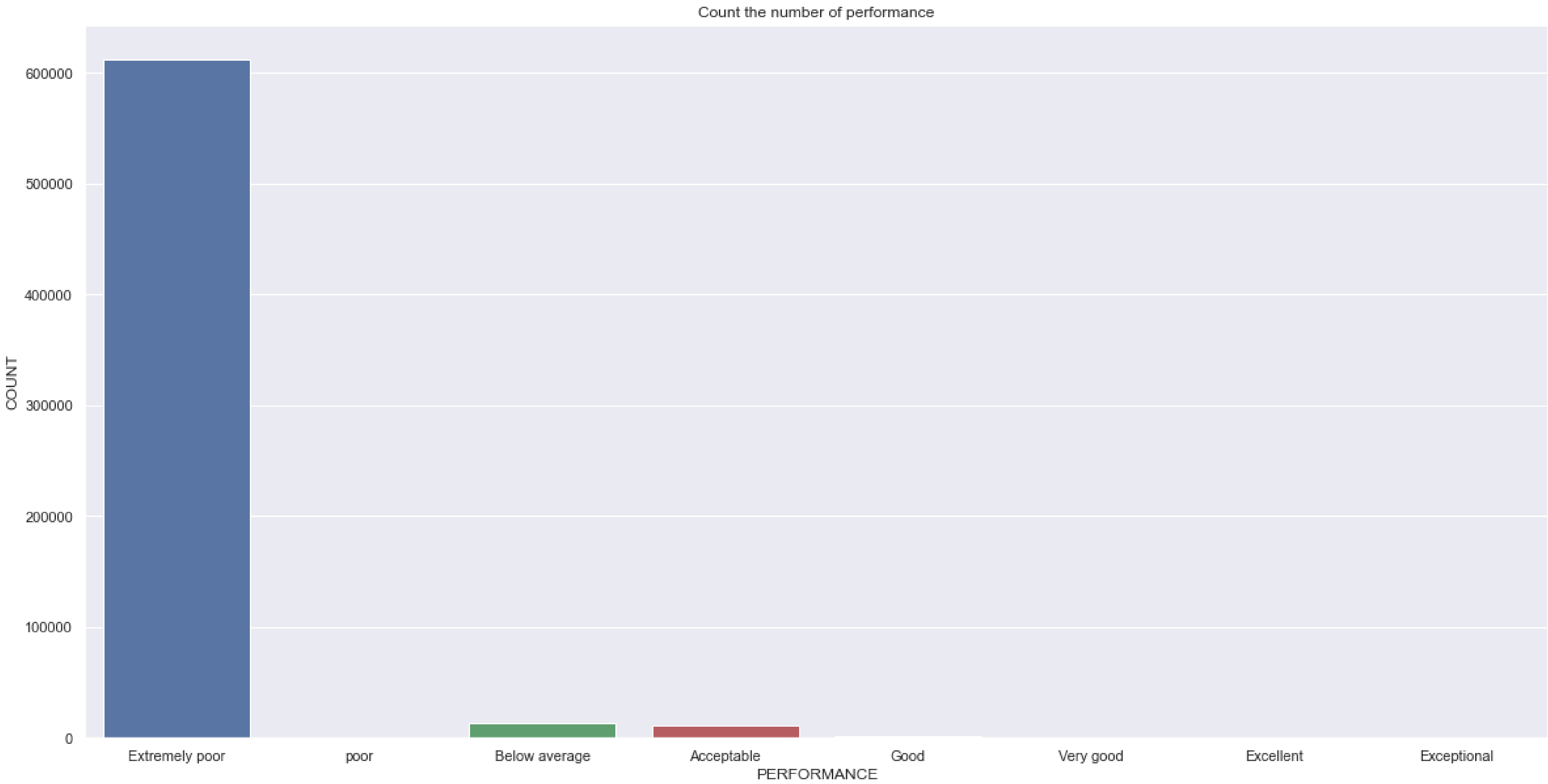
THE NUMBER OF AGE

the count number of each age



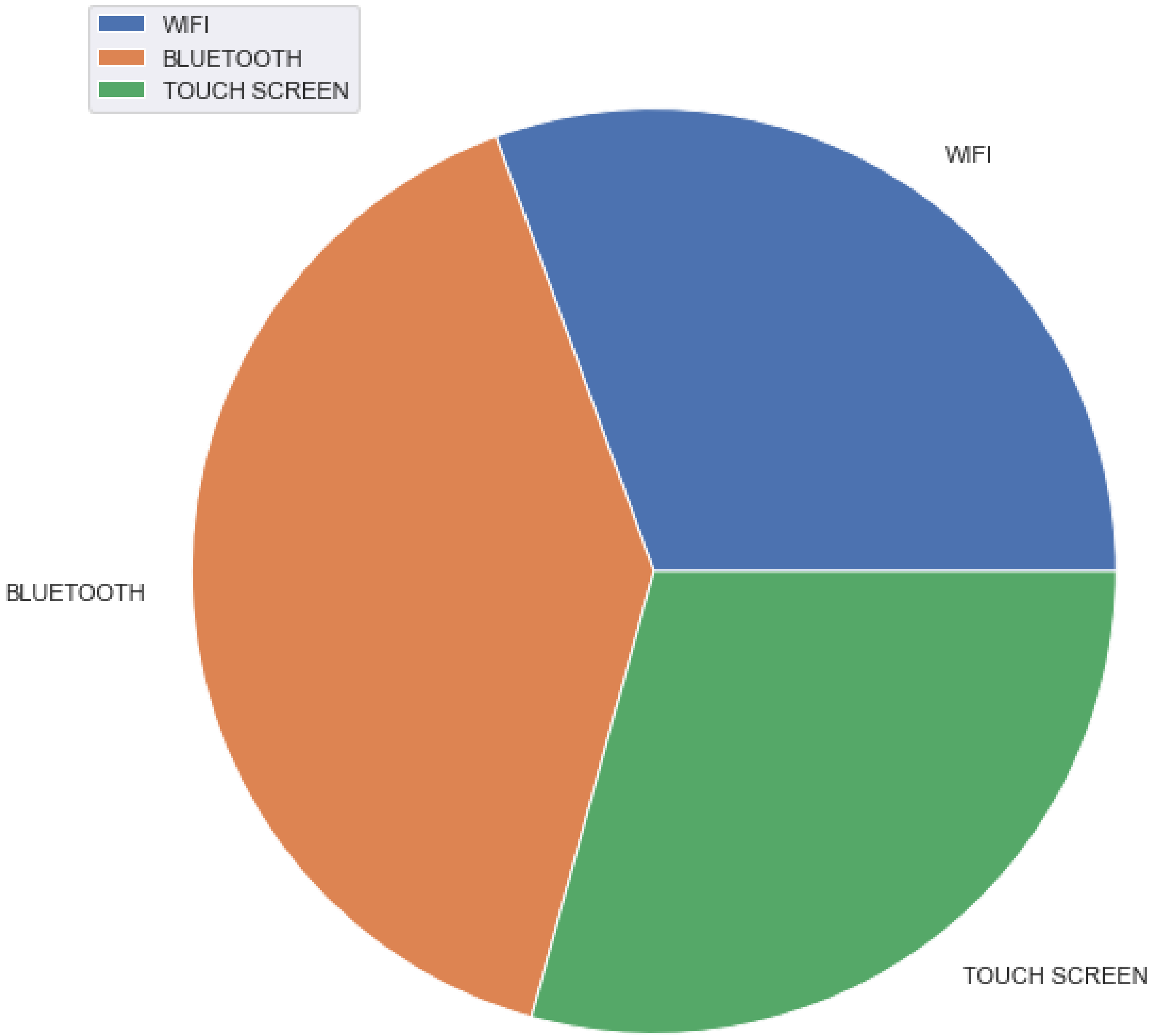
THE NUMBER OF PERFORMANCE

This graph shows the categorical of the performance columns



THE NUMBER OF SALES DEVICES BASED ON FEATURES

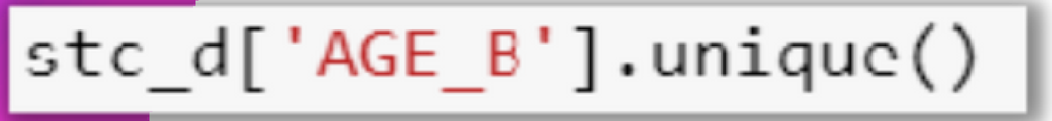
This graph shows the number of sales devices based on some features



DATA CLEANING

step 1

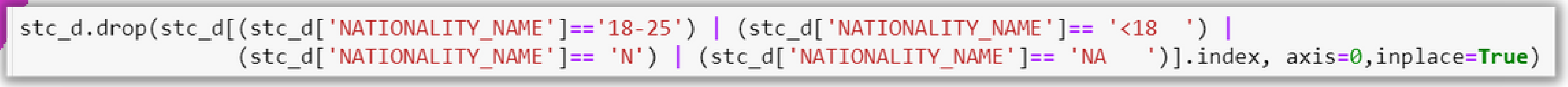
- finding unique values for all features



```
stc_d['AGE_B'].unique()
```

step 2

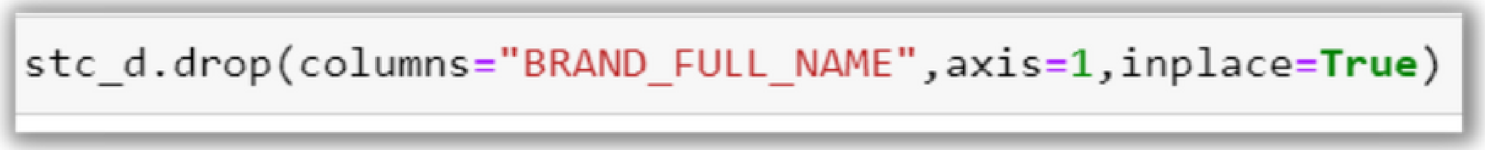
dropping unnecessary values



```
stc_d.drop(stc_d[(stc_d['NATIONALITY_NAME']=='18-25') | (stc_d['NATIONALITY_NAME']== '<18  ') |
                 (stc_d['NATIONALITY_NAME']== 'N') | (stc_d['NATIONALITY_NAME']== 'NA   ')].index, axis=0,inplace=True)
```

step 3

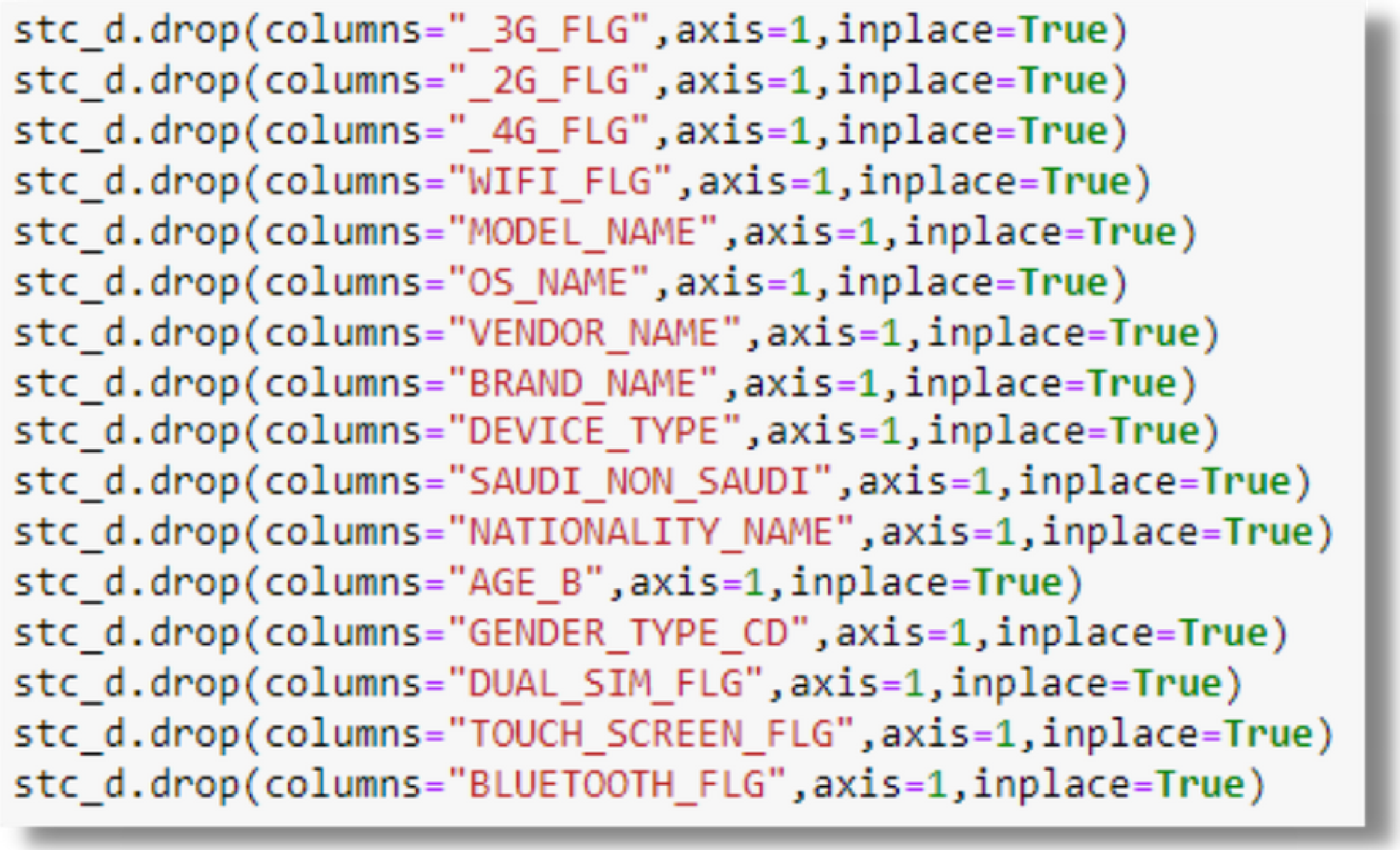
removing unnecessary cilumns



```
stc_d.drop(columns="BRAND_FULL_NAME",axis=1,inplace=True)
```

step 4

removing duplicated columns after encoding

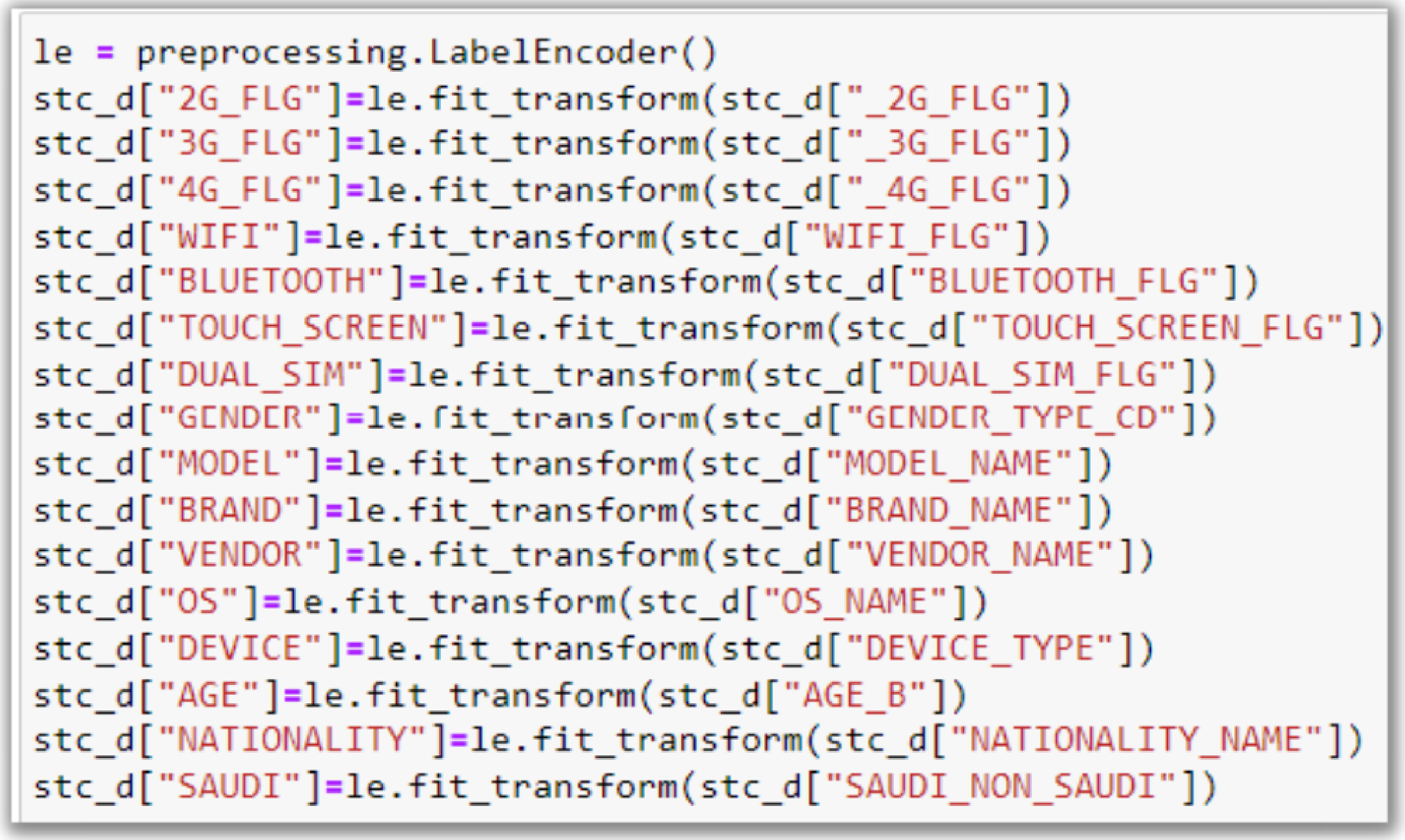


```
stc_d.drop(columns="_3G_FLG",axis=1,inplace=True)
stc_d.drop(columns="_2G_FLG",axis=1,inplace=True)
stc_d.drop(columns="_4G_FLG",axis=1,inplace=True)
stc_d.drop(columns="WIFI_FLG",axis=1,inplace=True)
stc_d.drop(columns="MODEL_NAME",axis=1,inplace=True)
stc_d.drop(columns="OS_NAME",axis=1,inplace=True)
stc_d.drop(columns="VENDOR_NAME",axis=1,inplace=True)
stc_d.drop(columns="BRAND_NAME",axis=1,inplace=True)
stc_d.drop(columns="DEVICE_TYPE",axis=1,inplace=True)
stc_d.drop(columns="SAUDI_NON_SAUDI",axis=1,inplace=True)
stc_d.drop(columns="NATIONALITY_NAME",axis=1,inplace=True)
stc_d.drop(columns="AGE_B",axis=1,inplace=True)
stc_d.drop(columns="GENDER_TYPE_CD",axis=1,inplace=True)
stc_d.drop(columns="DUAL_SIM_FLG",axis=1,inplace=True)
stc_d.drop(columns="TOUCH_SCREEN_FLG",axis=1,inplace=True)
stc_d.drop(columns="BLUETOOTH_FLG",axis=1,inplace=True)
```

DATA PREPROCESING

step 2

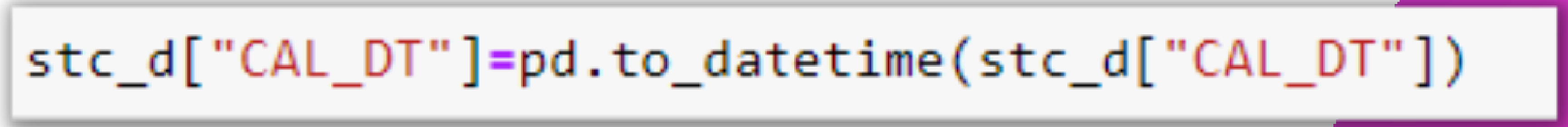
Using label encoder on the columns



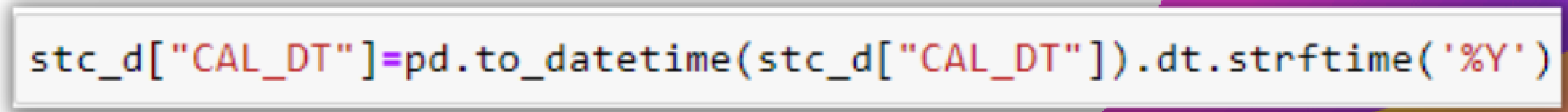
```
le = preprocessing.LabelEncoder()
stc_d["2G_FLG"]=le.fit_transform(stc_d["_2G_FLG"])
stc_d["3G_FLG"]=le.fit_transform(stc_d["_3G_FLG"])
stc_d["4G_FLG"]=le.fit_transform(stc_d["_4G_FLG"])
stc_d["WIFI"]=le.fit_transform(stc_d["WIFI_FLG"])
stc_d["BLUETOOTH"]=le.fit_transform(stc_d["BLUETOOTH_FLG"])
stc_d["TOUCH_SCREEN"]=le.fit_transform(stc_d["TOUCH_SCREEN_FLG"])
stc_d["DUAL_SIM"]=le.fit_transform(stc_d["DUAL_SIM_FLG"])
stc_d["GENDER"]=le.fit_transform(stc_d["GENDER_TYPE_CD"])
stc_d["MODEL"]=le.fit_transform(stc_d["MODEL_NAME"])
stc_d["BRAND"]=le.fit_transform(stc_d["BRAND_NAME"])
stc_d["VENDOR"]=le.fit_transform(stc_d["VENDOR_NAME"])
stc_d["OS"]=le.fit_transform(stc_d["OS_NAME"])
stc_d["DEVICE"]=le.fit_transform(stc_d["DEVICE_TYPE"])
stc_d["AGE"]=le.fit_transform(stc_d["AGE_B"])
stc_d["NATIONALITY"]=le.fit_transform(stc_d["NATIONALITY_NAME"])
stc_d["SAUDI"]=le.fit_transform(stc_d["SAUDI_NON_SAUDI"])
```

step 1

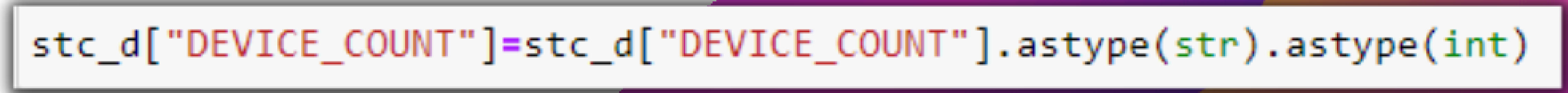
Changing column types



```
stc_d["CAL_DT"]=pd.to_datetime(stc_d["CAL_DT"])
```



```
stc_d["CAL_DT"]=pd.to_datetime(stc_d["CAL_DT"]).dt.strftime('%Y')
```



```
stc_d["DEVICE_COUNT"]=stc_d["DEVICE_COUNT"].astype(str).astype(int)
```

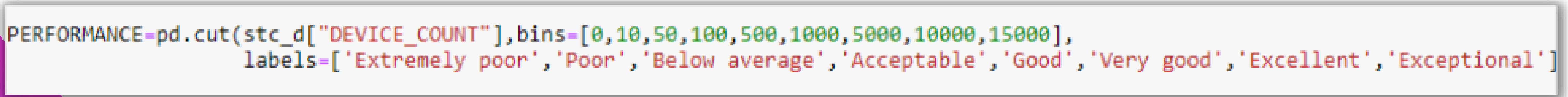
ML MODELS COMPARISON

stc

MODELS USED

We wanted to get the performance of the company in terms of device count, We created a new column for performance based on the count of devices with several bins to categorize the performance into categories such as :

- Extremely poor, poor, acceptable, good, very good, excellent and exceptional.



```
PERFORMANCE=pd.cut(stc_d["DEVICE_COUNT"],bins=[0,10,50,100,500,1000,5000,10000,15000],
                   labels=['Extremely poor','Poor','Below average','Acceptable','Good','Very good','Excellent','Exceptional']
```

comparison of model results

models	precision	recall	f1-score	support	accuracy
Logistic Regression	0.74	0.86	0.79	142801	0.86
Random Forest	0.95	0.95	0.95	142801	0.95
Decision Tree	0.78	0.86	0.79	142801	0.86
XGBoost	0.85	0.88	0.85	142801	0.88

comparison of tuning model results

models	precision	recall	f1-score	support	accuracy
Logistic Regression	0.74	0.86	0.79	142801	0.85
Random Forest	0.74	0.86	0.79	142801	0.86
Decision Tree	0.78	0.86	0.79	142801	0.86
XGBoost	0.80	0.86	0.80	142801	0.86

FUTURE WORK

- Use more datasets
- Try oversampling
- Try hyperparameter tuning using Randomized search

References

- **STC** https://www.stc.com.sa/
- **Vision 2030** https://www.vision2030.gov.sa
- **STC**: The Change Management Process and the Saudi 2030 vision. | LinkedIn

THANK YOU