Task 3

Build a decision tree classifier to predict whether a customer will purchase a product or service based on their demographic and behavioral data. Use a dataset such as the Bank Marketing dataset from the UCI Machine Learning Repository.

About the dataset:

The data is related with direct marketing campaigns (phone calls) of a Portuguese banking institution. The classification goal is to predict if the client will subscribe a term deposit (variable y).

+ Code + Text

Importing the necessary libraries

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

Reading the dataset

```
df = pd.read_csv(r"/content/bank-additional-full.csv",delimiter=';')
```

```
df.head(5)
```

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	...	campaign	pdays
0	56	housemaid	married	basic.4y	no	no	no	telephone	may	mon	...	1	999
1	57	services	married	high.school	unknown	no	no	telephone	may	mon	...	1	999
2	37	services	married	high.school	no	yes	no	telephone	may	mon	...	1	999
3	40	admin.	married	basic.6y	no	no	no	telephone	may	mon	...	1	999
4	56	services	married	high.school	no	no	yes	telephone	may	mon	...	1	999

5 rows × 21 columns

```
df.rename(columns={'y':'subscribed_deposit'}, inplace=True)
```

Details about the dataset

```
df.shape
```

```
(41188, 21)
```

```
df.columns
```

```
Index(['age', 'job', 'marital', 'education', 'default', 'housing', 'loan',
       'contact', 'month', 'day_of_week', 'duration', 'campaign', 'pdays',
       'previous', 'poutcome', 'emp.var.rate', 'cons.price.idx',
       'cons.conf.idx', 'euribor3m', 'nr.employed', 'subscribed_deposit'],
      dtype='object')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41188 entries, 0 to 41187
Data columns (total 21 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
```

```
 0   age                41188 non-null  int64
 1   job                41188 non-null  object
 2   marital            41188 non-null  object
 3   education          41188 non-null  object
 4   default            41188 non-null  object
 5   housing            41188 non-null  object
 6   loan               41188 non-null  object
 7   contact            41188 non-null  object
 8   month              41188 non-null  object
 9   day_of_week        41188 non-null  object
 10  duration           41188 non-null  int64
 11  campaign           41188 non-null  int64
 12  pdays              41188 non-null  int64
 13  previous           41188 non-null  int64
 14  poutcome           41188 non-null  object
 15  emp.var.rate       41188 non-null  float64
 16  cons.price.idx     41188 non-null  float64
 17  cons.conf.idx      41188 non-null  float64
 18  euribor3m          41188 non-null  float64
 19  nr.employed        41188 non-null  float64
 20  subscribed_deposit  41188 non-null  object
dtypes: float64(5), int64(5), object(11)
memory usage: 6.6+ MB
```

```
df.describe()
```

	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.i
count	41188.00000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.0000
mean	40.02406	258.285010	2.567593	962.475454	0.172963	0.081886	93.575664	-40.5026
std	10.42125	259.279249	2.770014	186.910907	0.494901	1.570960	0.578840	4.6281
min	17.00000	0.000000	1.000000	0.000000	0.000000	-3.400000	92.201000	-50.8000
25%	32.00000	102.000000	1.000000	999.000000	0.000000	-1.800000	93.075000	-42.7000
50%	38.00000	180.000000	2.000000	999.000000	0.000000	1.100000	93.749000	-41.8000
75%	47.00000	319.000000	3.000000	999.000000	0.000000	1.400000	93.994000	-36.4000
max	98.00000	4918.000000	56.000000	999.000000	7.000000	1.400000	94.767000	-26.9000

Checking for null/missing values in the dataset

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

	0
age	0
job	0
marital	0
education	0
default	0
housing	0
loan	0
contact	0
month	0
day_of_week	0
duration	0
campaign	0
pdays	0
previous	0
poutcome	0
emp.var.rate	0
cons.price.idx	0
cons.conf.idx	0
euribor3m	0
nr.employed	0
subscribed_deposit	0

dtype: int64

Checking for duplicate values in the dataset

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

```
12
```

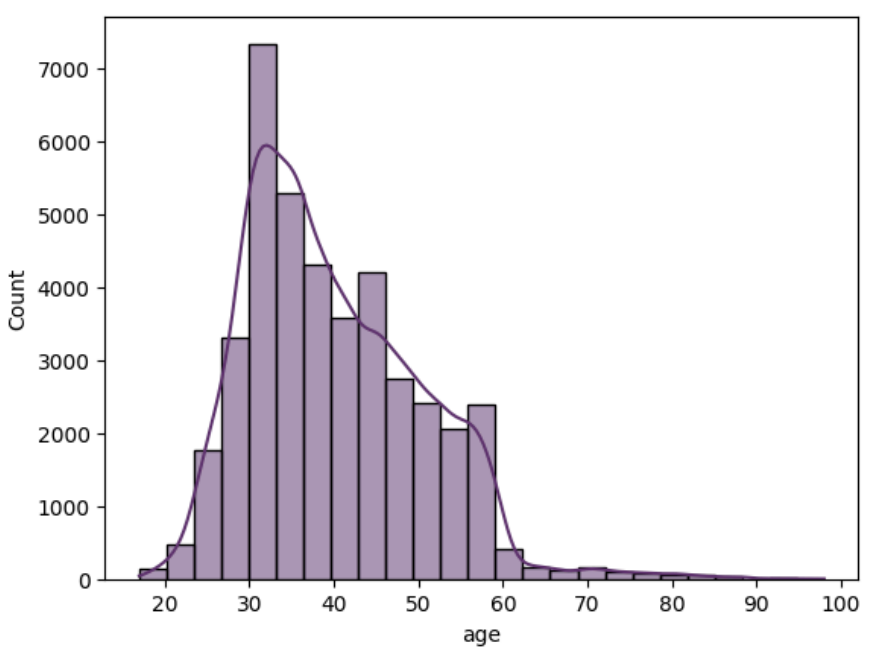
```
df.drop_duplicates(inplace=True)
```

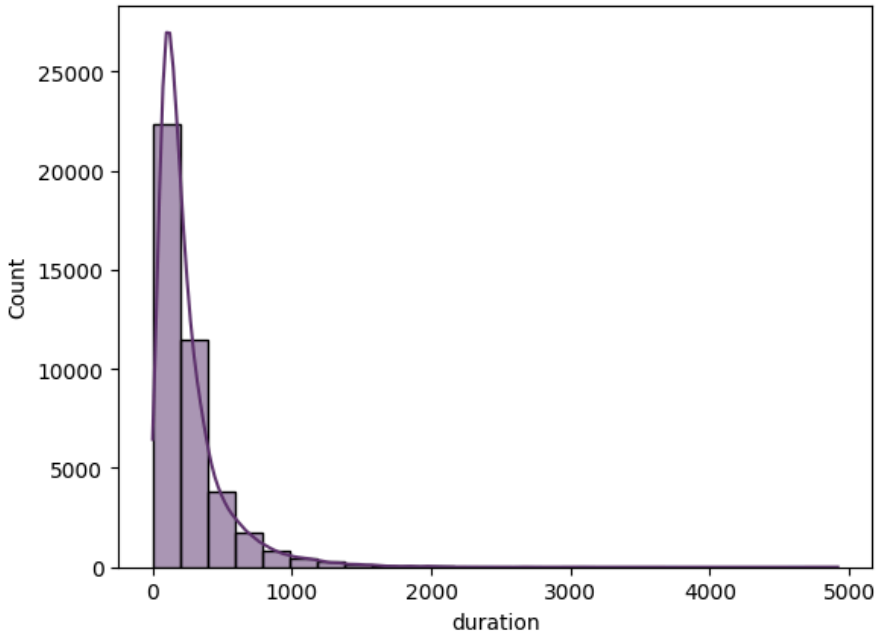
```
df.duplicated().sum()
```

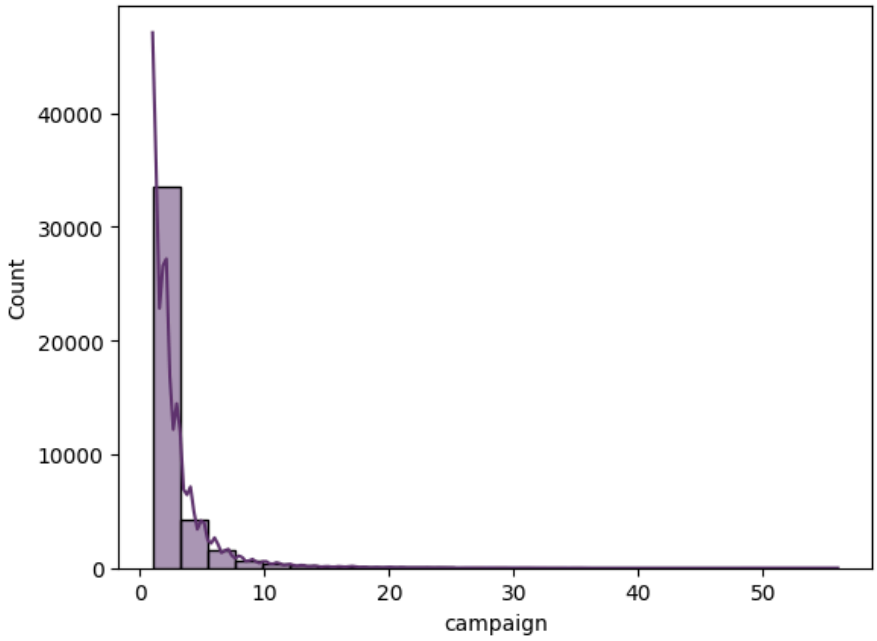
```
0
```

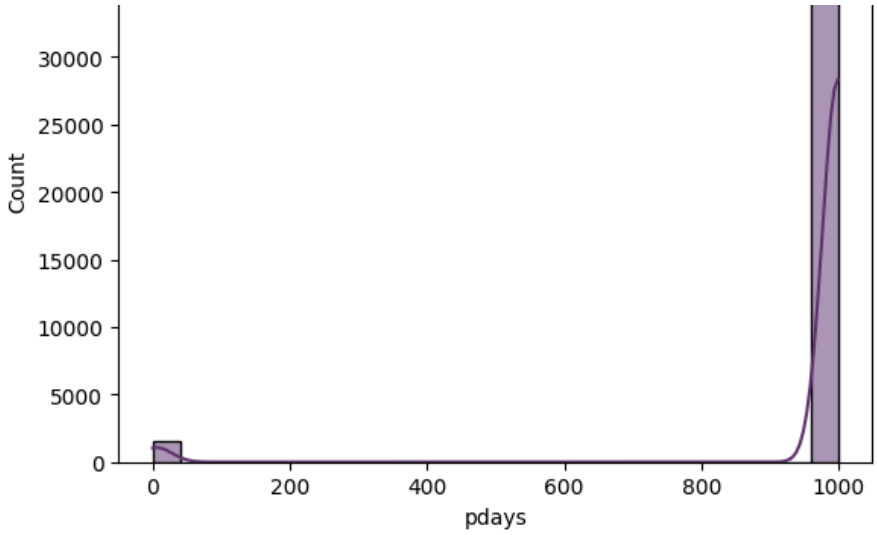
Visualizing numerical columns using histogram

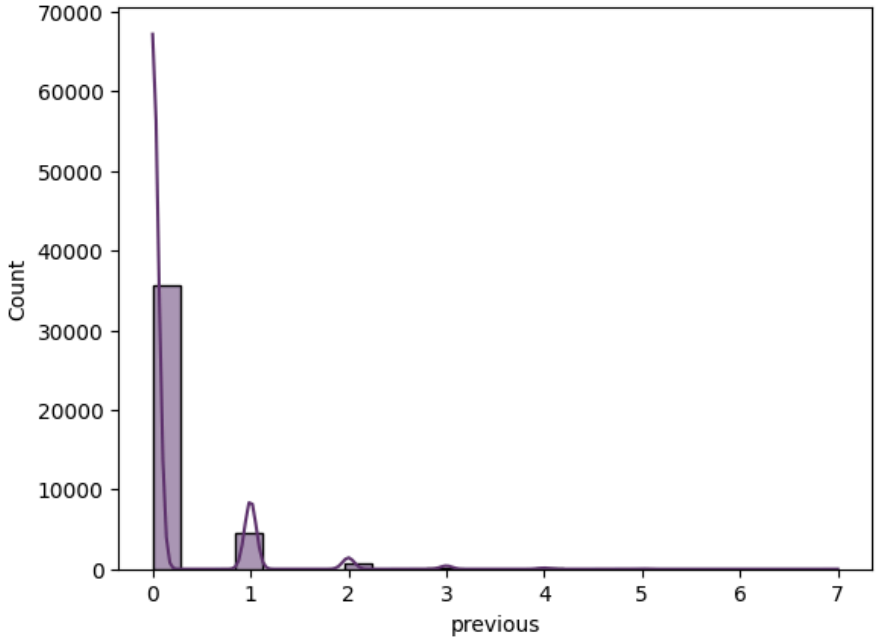
```
df_obj= df.select_dtypes(include='object').columns

df_num= df.select_dtypes(exclude='object').columns


for feature in df_num:
    sns.histplot(x=feature,data=df,bins=25,kde=True,color='#5f366e')
    plt.show()
```

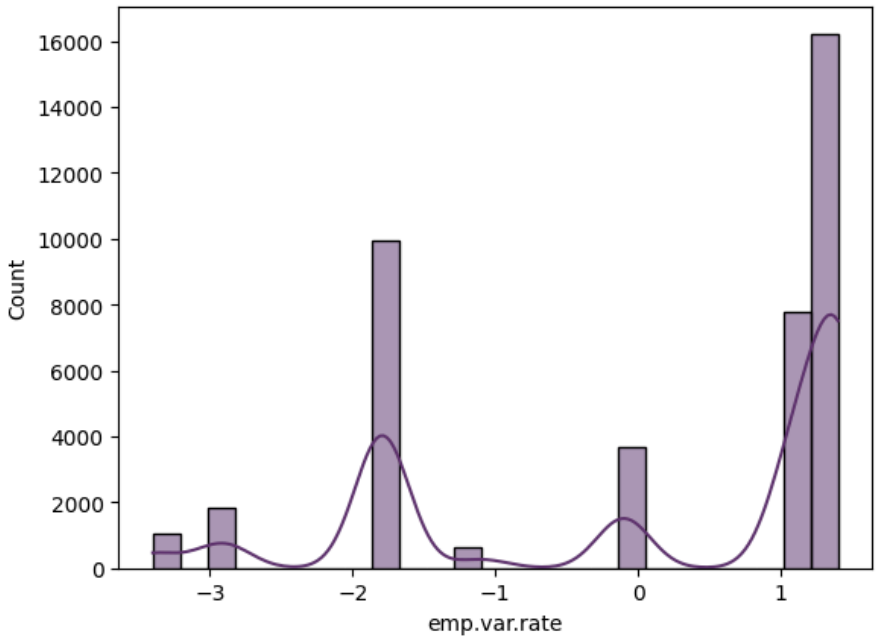

Count
age

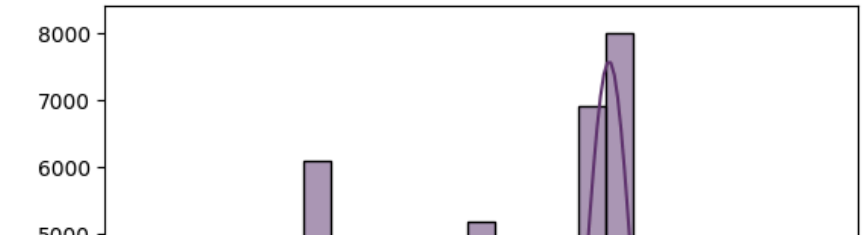

Count
duration

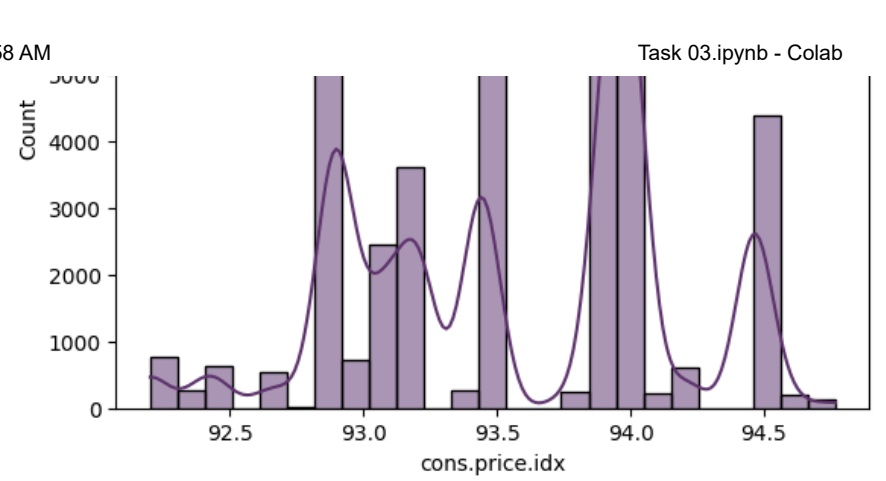

Count
campaign

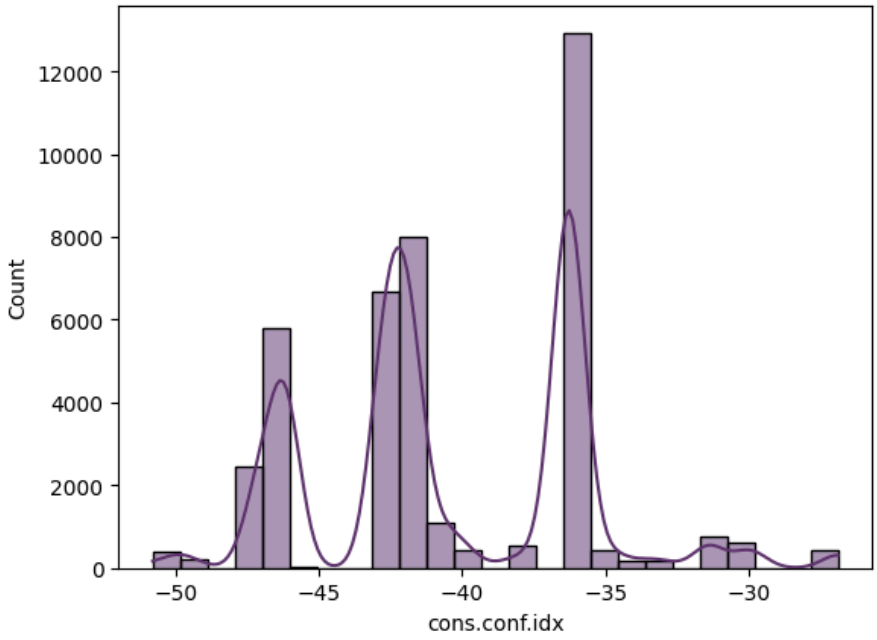

Count
pdays
0
5000
10000
15000
20000
25000
30000
0
200
400
600
800
1000

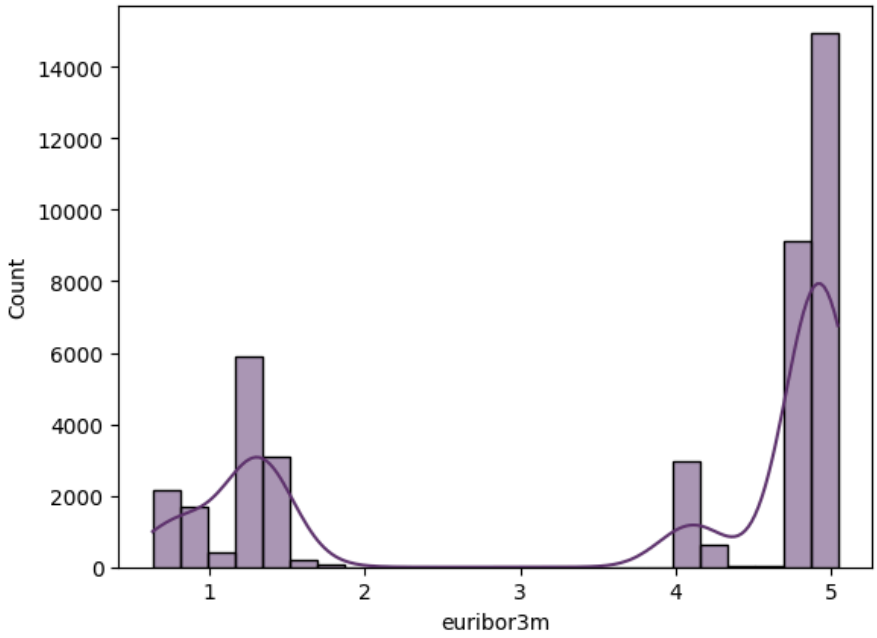

Count
previous
0
10000
20000
30000
40000
50000
60000
70000
0
1
2
3
4
5
6
7

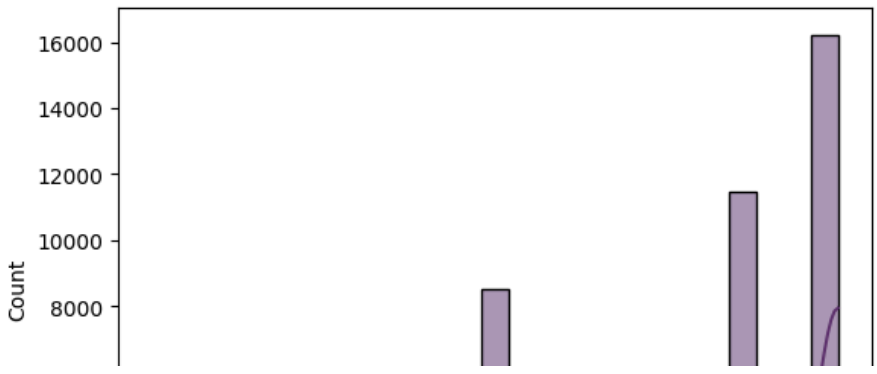

Count
emp.var.rate
0
2000
4000
6000
8000
10000
12000
14000
16000
−3
−2
−1
0
1

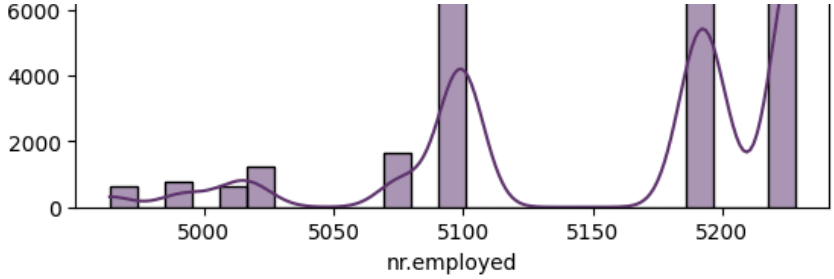

8000
7000
6000


Count
4000
3000
2000
1000
0
92.5
93.0
93.5
94.0
94.5
cons.price.idx


Count
12000
10000
8000
6000
4000
2000
0
−50
−45
−40
−35
−30
cons.conf.idx

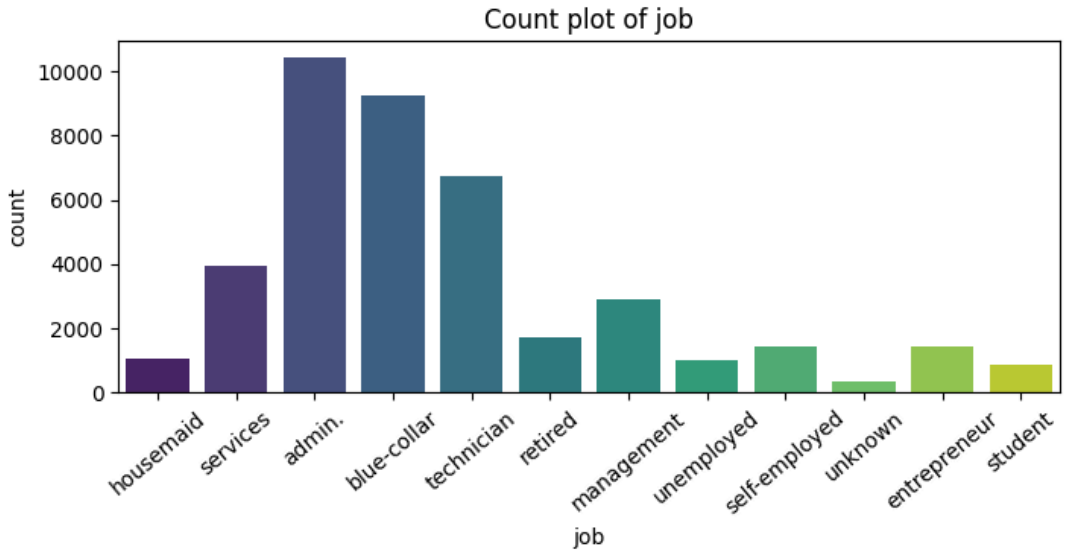

Count
14000
12000
10000
8000
6000
4000
2000
0
1
2
3
4
5
euribor3m

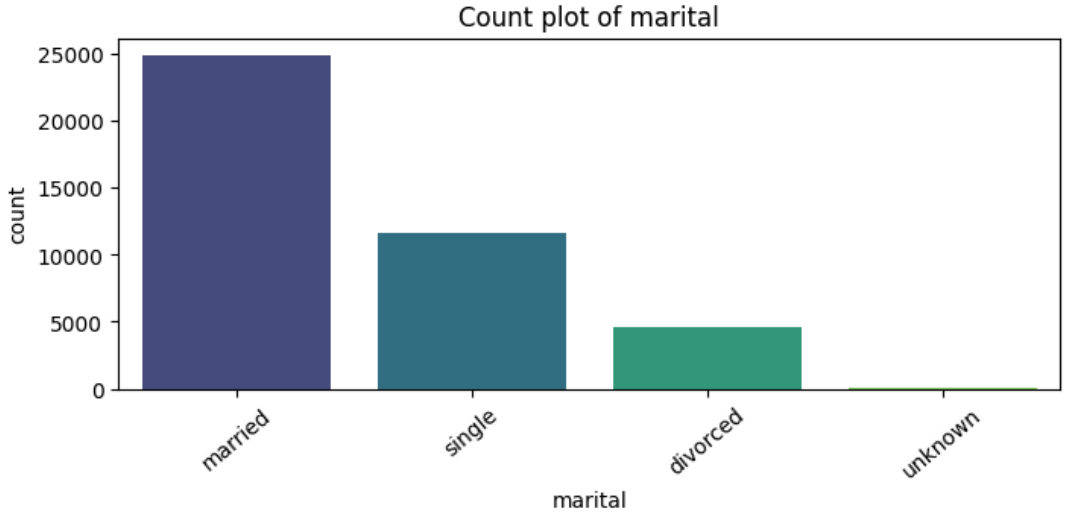

Count
16000
14000
12000
10000
8000

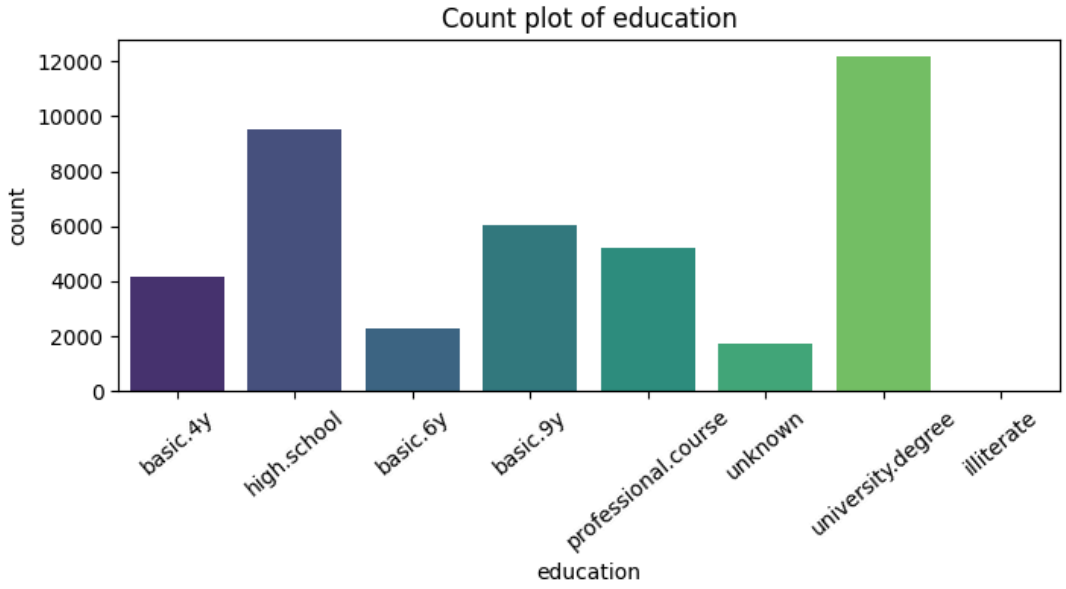


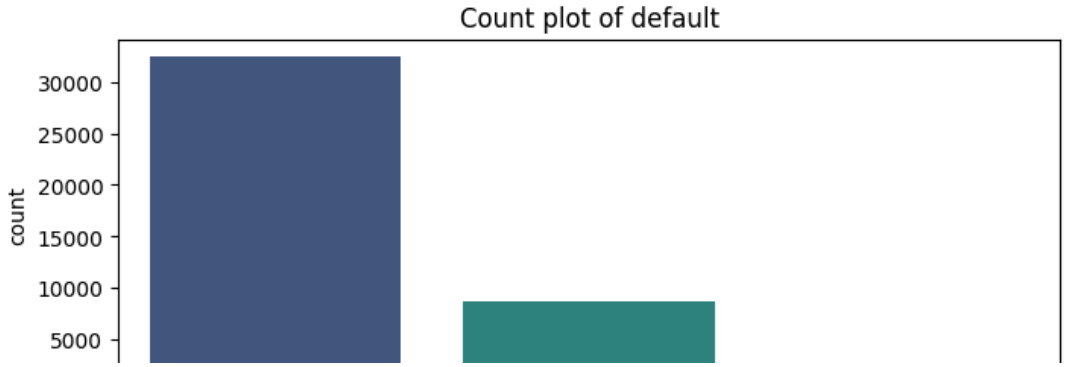
Visualizing categorical columns using bar graphs

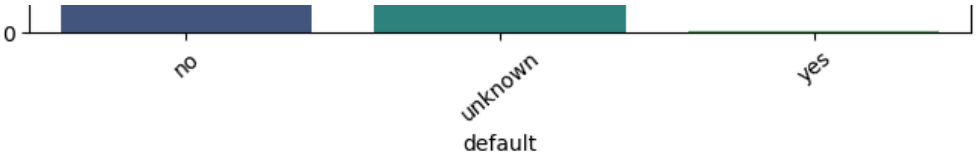
```
for feature in df_obj:
    plt.figure(figsize=(8,3))
    plt.title(f"Count plot of {feature}")
    sns.countplot(x=feature,data=df,palette='viridis')
    plt.xticks(rotation=40)
    plt.show()
```

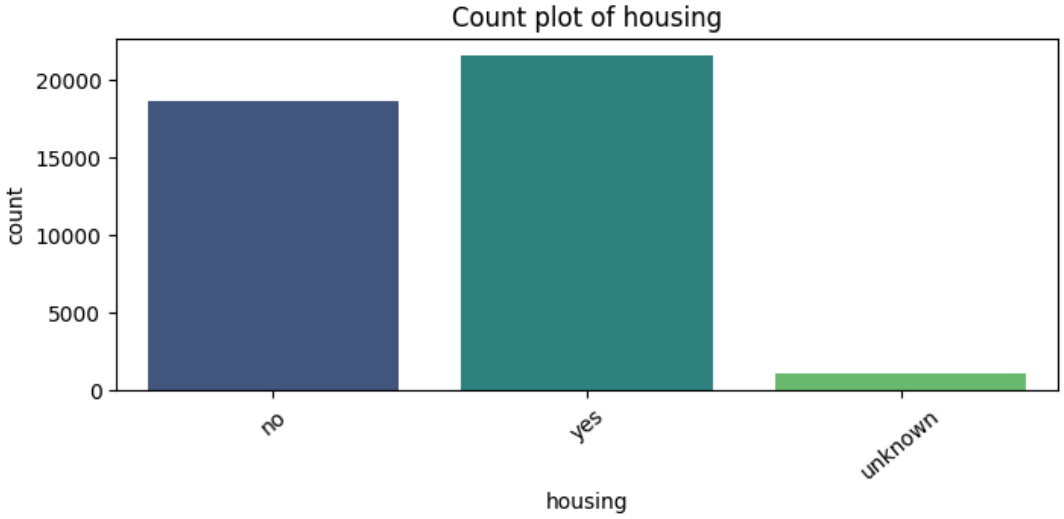

Count plot of job
count
job
housemaid
services
admin.
blue-collar
technician
retired
management
unemployed
self-employed
unknown
entrepreneur
student

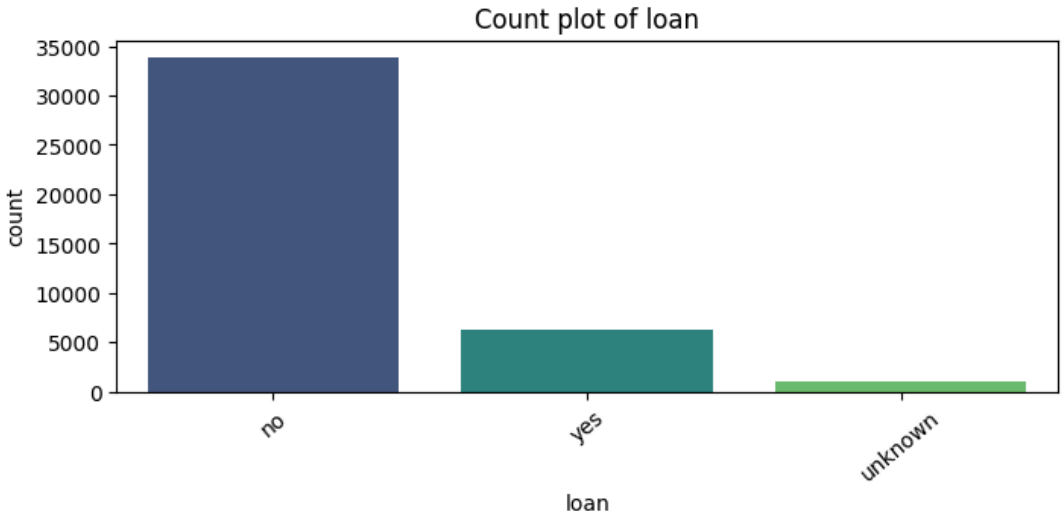

Count plot of marital
count
marital
married
single
divorced
unknown

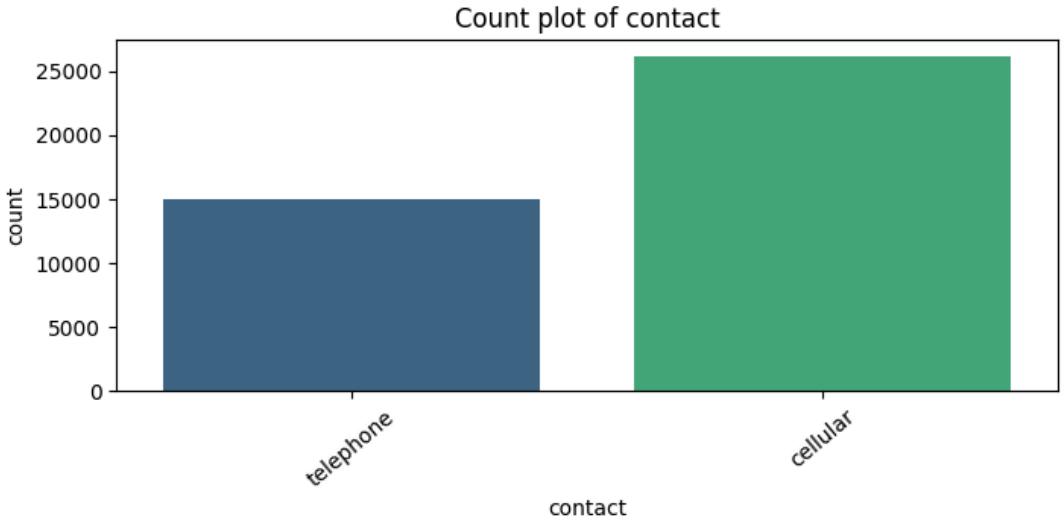

Count plot of education
count
education
basic.4y
high.school
basic.6y
basic.9y
professional.course
unknown
university.degree
illiterate

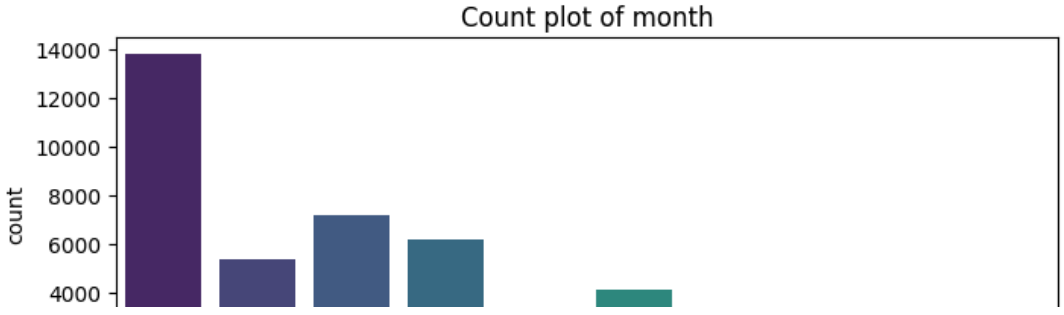

Count plot of default
count

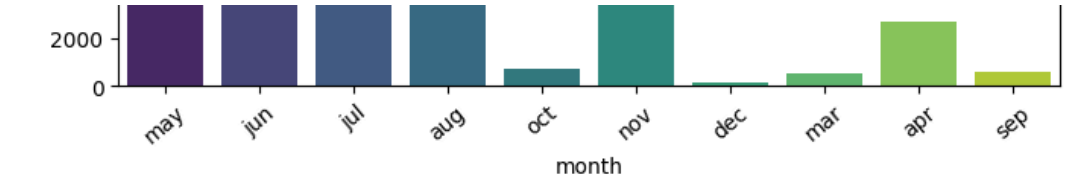

0
no
unknown
yes
default

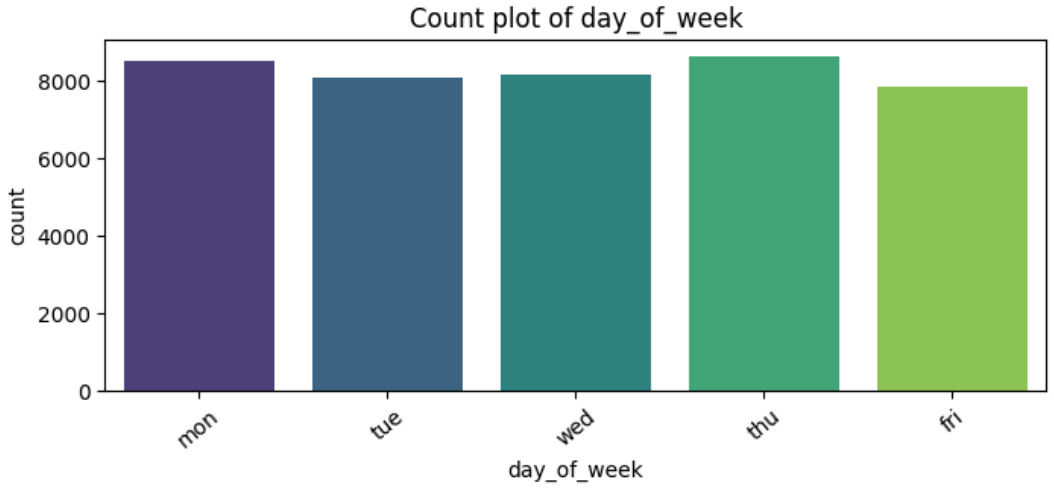

Count plot of housing
20000
15000
10000
5000
0
count
no
yes
unknown
housing

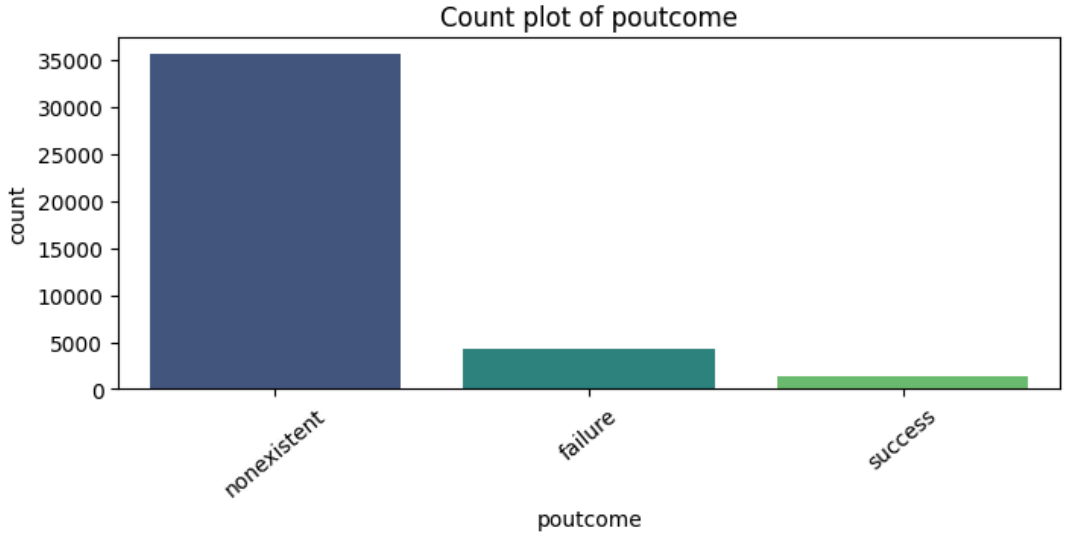

Count plot of loan
35000
30000
25000
20000
15000
10000
5000
0
count
no
yes
unknown
loan

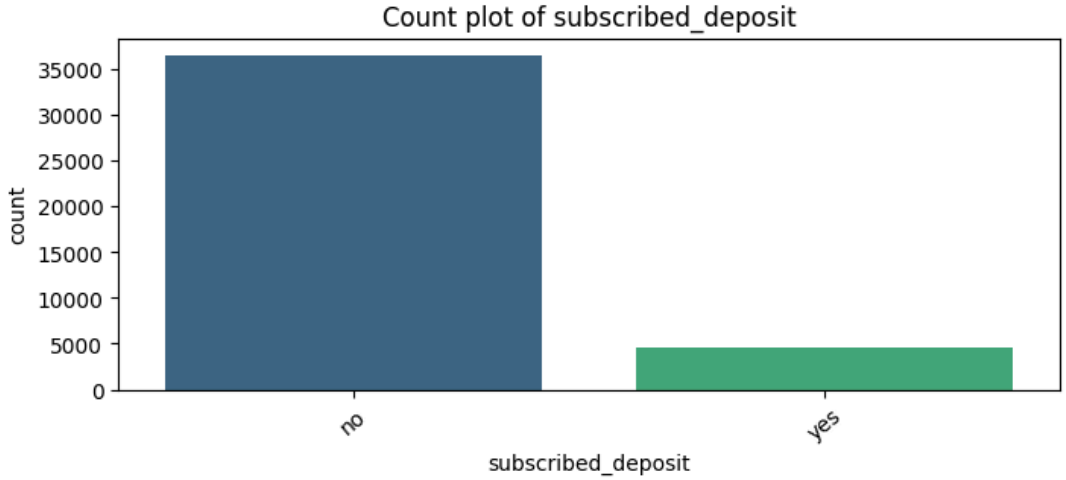

Count plot of contact
25000
20000
15000
10000
5000
0
count
telephone
cellular
contact


Count plot of month
14000
12000
10000
8000
6000
4000
count









Observations & Insights:

In the Job Column, we have seen most of the clients are working as 'admin'.

In the marital Column, we have seen most of the clients are married.

In the education Column, we have seen most of the clients are having 'university.degree' as education.

In the default Column, we have seen most of the clients are having 'no' credit as default.

In the housing Column, we have seen most of the clients are taking housing loan.

In the loan Column, we have seen most of the clients are not taking personal loan.

In the contact Column, we have seen most of the clients are choosen cellular as contact.

In the month Column, we have seen most of the clients are contacted in the 'may' month.

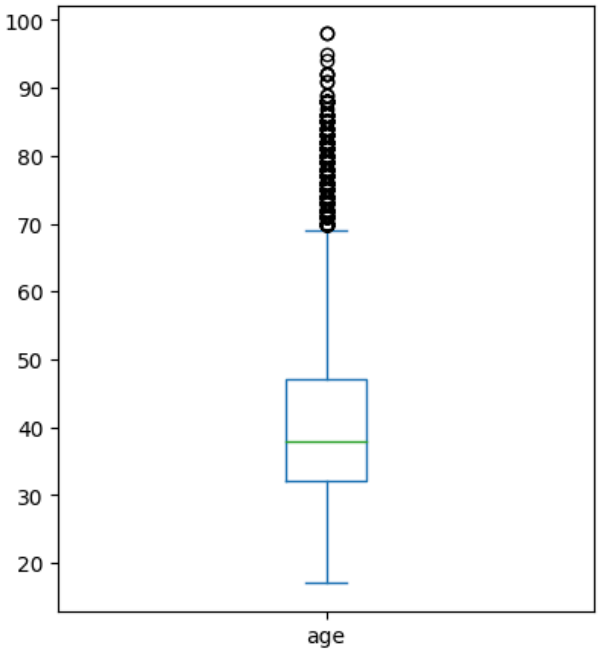
In the day_of_week Column, we have seen most of the clients are contacted in 'thursday'.

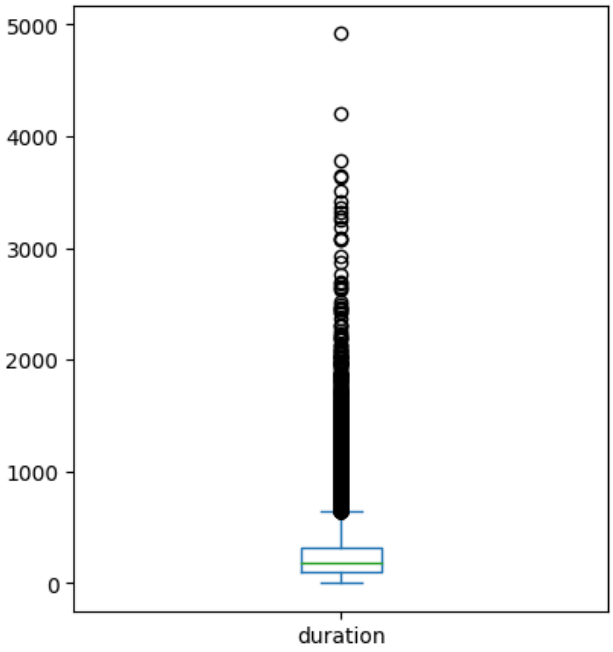
In the poutcome Column, we have seen the result of most of the previous market campaign is 'nonexistent'.

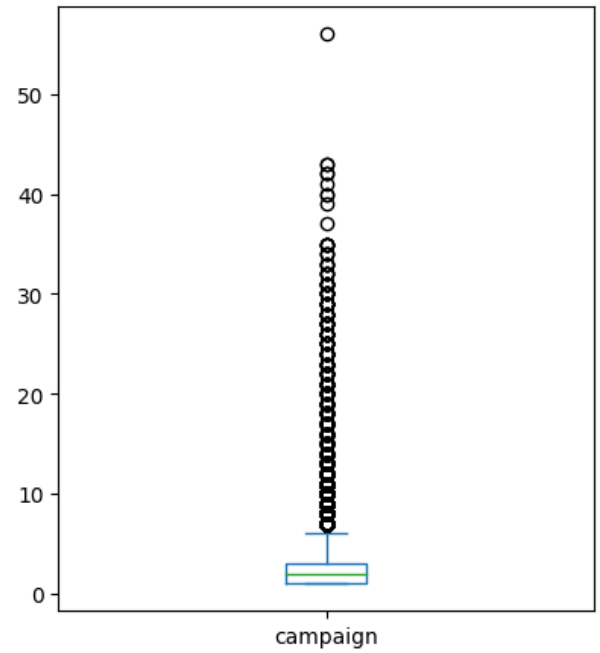
In the target column , we have seen most of the clients are not subscribed a term deposit.

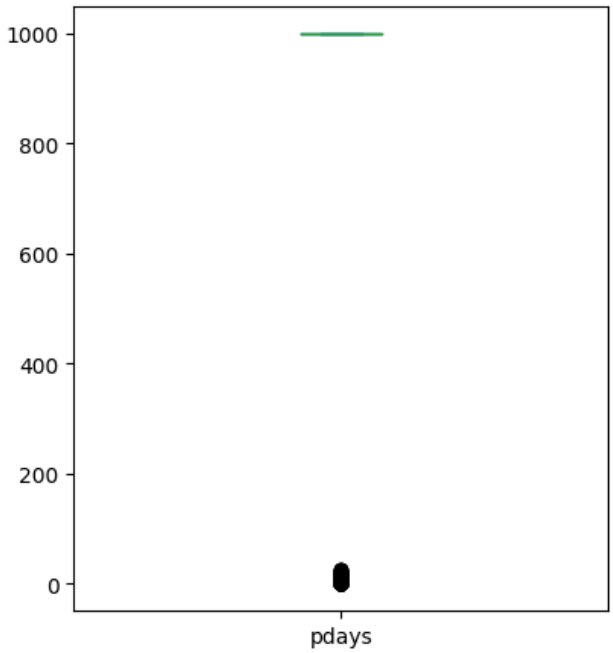
Checking for outliers

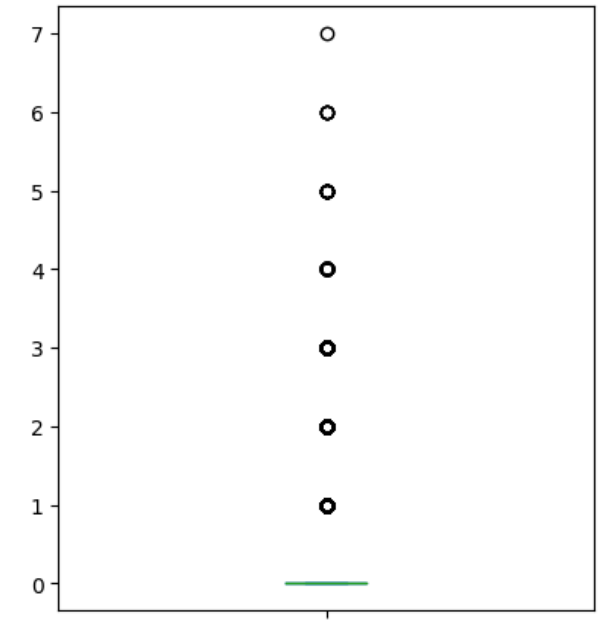
```
df.plot(kind='box', subplots=True, layout=(5,2), figsize=(10,30))
plt.show()
```

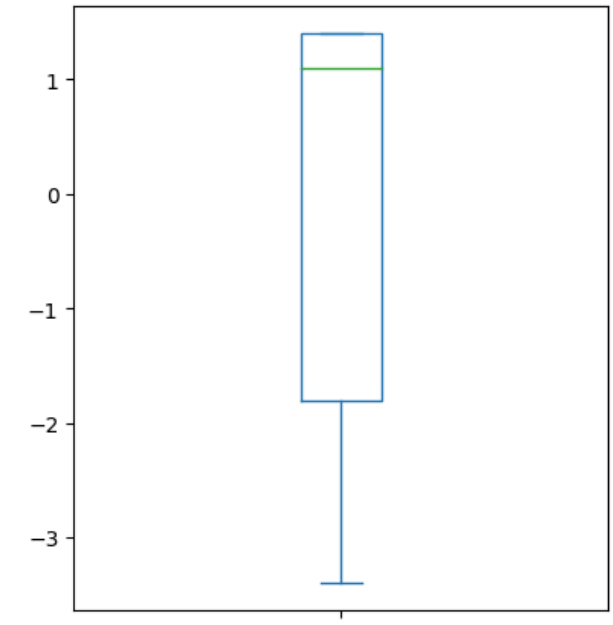

100
90
80
70
60
50
40
30
20
age


5000
4000
3000
2000
1000
0
duration


50
40
30
20
10
0
campaign


1000
800
600
400
200
0
pdays


7
6
5
4
3
2
1
0


1
0
−1
−2
−3