```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')

%matplotlib inline
```

```python
df = pd.read_csv('/Users/aaryanbabuta/Documents/Prodigy DS Internship June 2024/bank+marketing/bank-additional/bank-additional.csv',delimiter=';')
df.rename(columns={'y':'deposit'}, inplace=True)
df.head()
```

```
   age          job  marital          education default  housing
loan  \
0   30  blue-collar  married           basic.9y      no      yes
no
1   39     services   single        high.school      no       no
no
2   25     services  married        high.school      no      yes
no
3   38     services  married           basic.9y      no  unknown
unknown
4   47       admin.  married  university.degree      no      yes
no

     contact month day_of_week  ...  campaign  pdays  previous
poutcome  \
0   cellular   may         fri  ...         2    999         0
nonexistent
1  telephone   may         fri  ...         4    999         0
nonexistent
2  telephone   jun         wed  ...         1    999         0
nonexistent
3  telephone   jun         fri  ...         3    999         0
nonexistent
4   cellular   nov         mon  ...         1    999         0
nonexistent

  emp.var.rate  cons.price.idx  cons.conf.idx  euribor3m  nr.employed
deposit
0         -1.8          92.893          -46.2      1.313       5099.1
no
1          1.1          93.994          -36.4      4.855       5191.0
no
2          1.4          94.465          -41.8      4.962       5228.1
no
```

```
3          1.4          94.465          -41.8      4.959       5228.1
no
4         -0.1          93.200          -42.0      4.191       5195.8
no

[5 rows x 21 columns]
```

```
df.head()
```

```
   age          job  marital          education default  housing
loan  \
0   30  blue-collar  married           basic.9y      no      yes
no
1   39     services   single        high.school      no       no
no
2   25     services  married        high.school      no      yes
no
3   38     services  married           basic.9y      no  unknown
unknown
4   47       admin.  married  university.degree      no      yes
no

     contact month day_of_week  ...  campaign  pdays  previous
poutcome  \
0   cellular   may         fri  ...         2    999         0
nonexistent
1  telephone   may         fri  ...         4    999         0
nonexistent
2  telephone   jun         wed  ...         1    999         0
nonexistent
3  telephone   jun         fri  ...         3    999         0
nonexistent
4   cellular   nov         mon  ...         1    999         0
nonexistent

  emp.var.rate  cons.price.idx  cons.conf.idx  euribor3m  nr.employed
deposit
0         -1.8          92.893          -46.2      1.313       5099.1
no
1          1.1          93.994          -36.4      4.855       5191.0
no
2          1.4          94.465          -41.8      4.962       5228.1
no
3          1.4          94.465          -41.8      4.959       5228.1
no
4         -0.1          93.200          -42.0      4.191       5195.8
no

[5 rows x 21 columns]
```

```
df.shape
```

```
(4119, 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', 'deposit'],
      dtype='object')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4119 entries, 0 to 4118
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   age             4119 non-null   int64
 1   job             4119 non-null   object
 2   marital         4119 non-null   object
 3   education       4119 non-null   object
 4   default         4119 non-null   object
 5   housing         4119 non-null   object
 6   loan            4119 non-null   object
 7   contact         4119 non-null   object
 8   month           4119 non-null   object
 9   day_of_week     4119 non-null   object
 10  duration        4119 non-null   int64
 11  campaign        4119 non-null   int64
 12  pdays           4119 non-null   int64
 13  previous        4119 non-null   int64
 14  poutcome        4119 non-null   object
 15  emp.var.rate    4119 non-null   float64
 16  cons.price.idx  4119 non-null   float64
 17  cons.conf.idx   4119 non-null   float64
 18  euribor3m       4119 non-null   float64
 19  nr.employed     4119 non-null   float64
 20  deposit         4119 non-null   object
dtypes: float64(5), int64(5), object(11)
memory usage: 675.9+ KB
```

```
df.dtypes.value_counts()
```

```
object     11
int64       5
float64     5
Name: count, dtype: int64
```

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

```
0
```

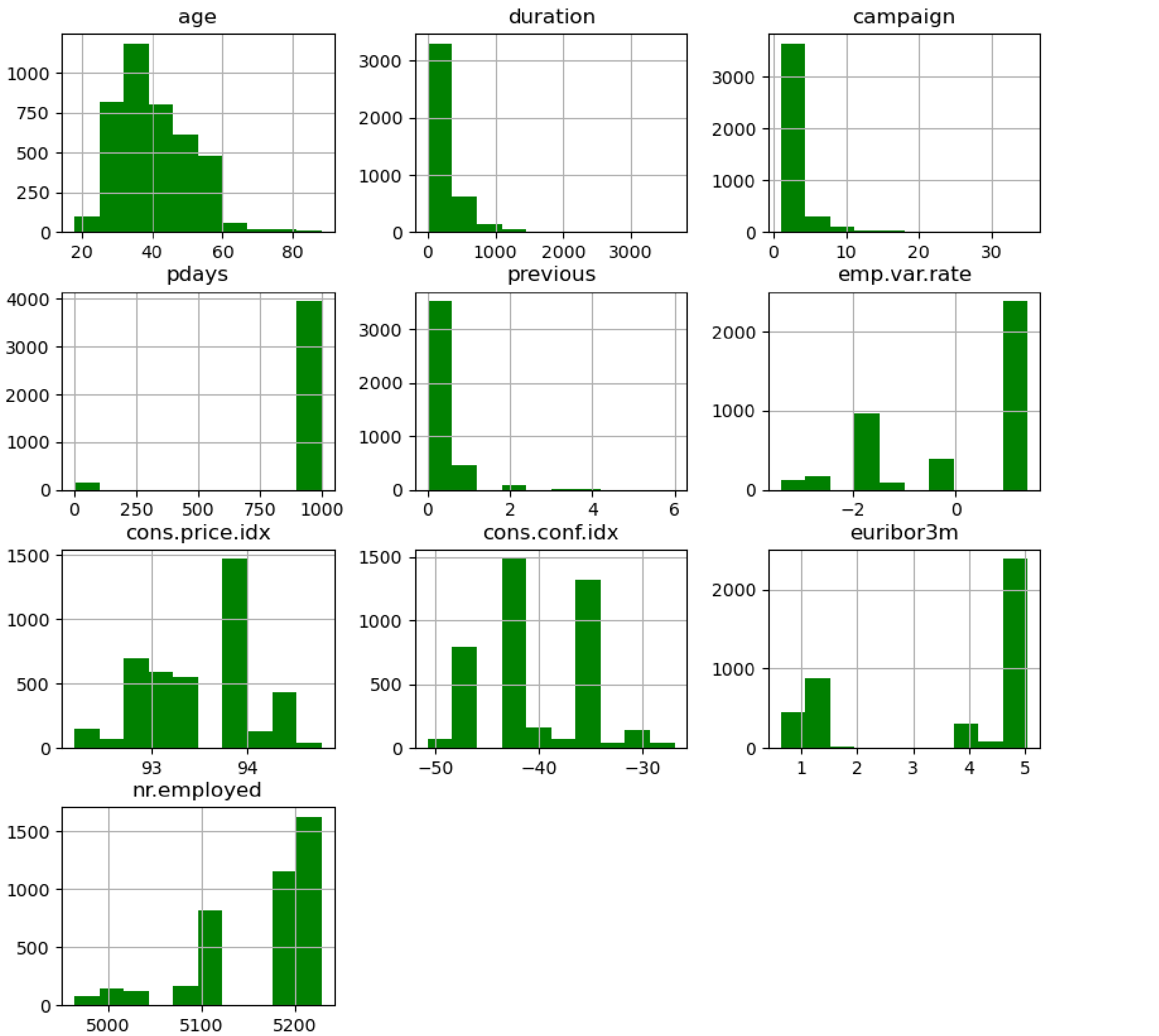
```
df.isna().sum().any()
```

```
False
```

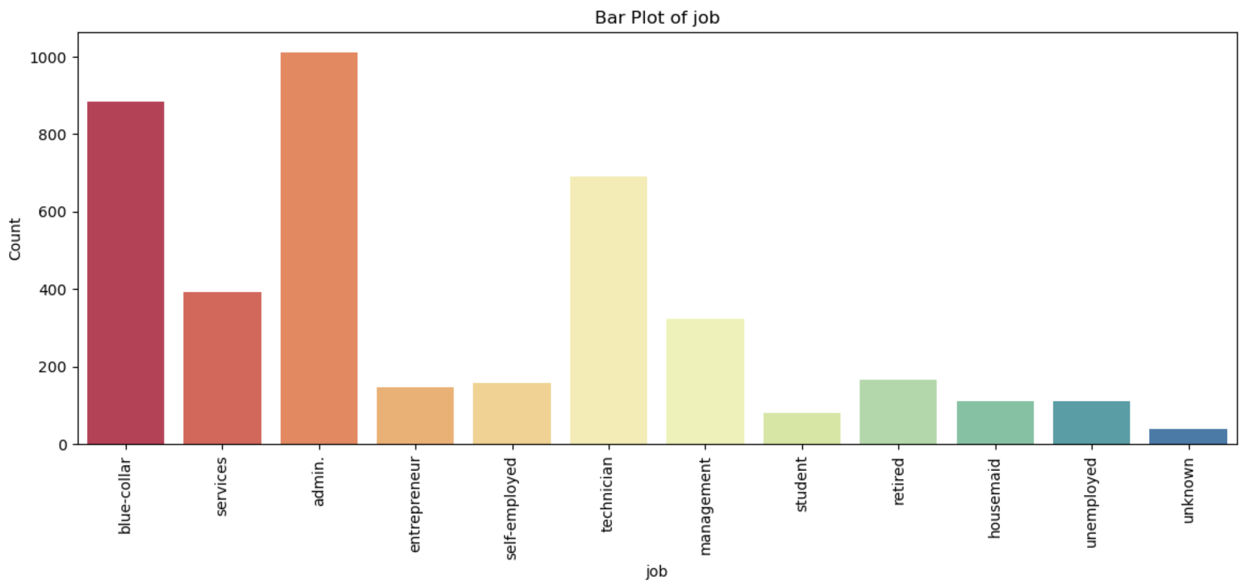
```
cat_cols = df.select_dtypes(include='object').columns
num_cols = df.select_dtypes(exclude='object').columns

print(cat_cols,"\n")
print(num_cols)
```

```
Index(['job', 'marital', 'education', 'default', 'housing', 'loan',
'contact',
       'month', 'day_of_week', 'poutcome', 'deposit'],
      dtype='object')

Index(['age', 'duration', 'campaign', 'pdays', 'previous',
'emp.var.rate',
       'cons.price.idx', 'cons.conf.idx', 'euribor3m', 'nr.employed'],
      dtype='object')
```

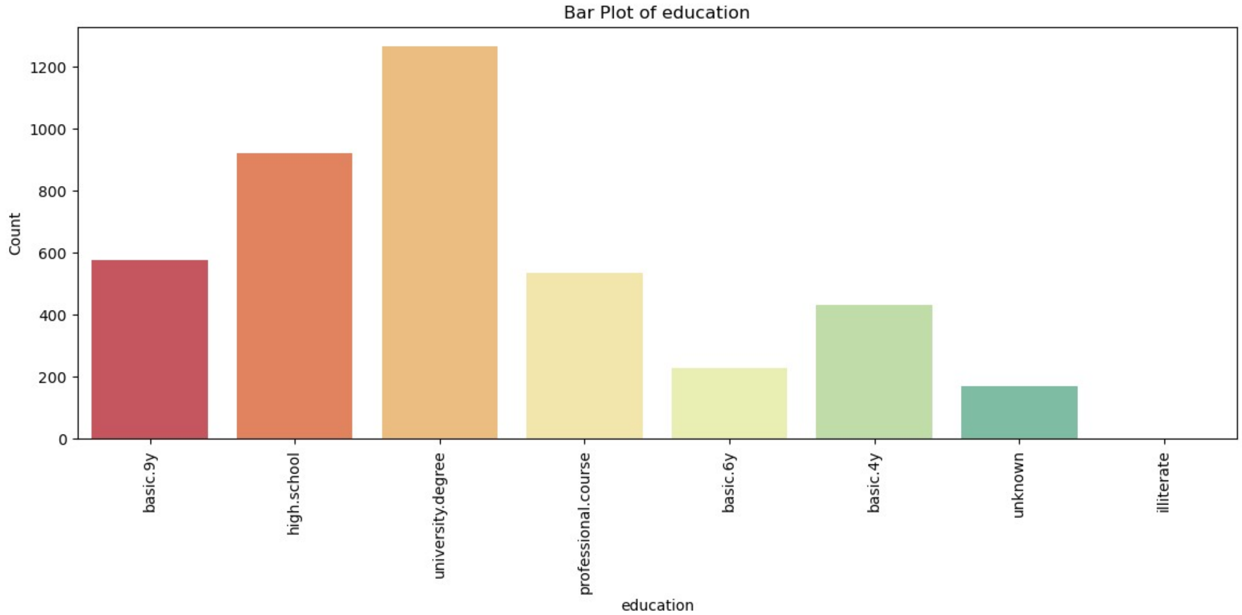
```
df.hist(figsize=(10,10),color='Green')
plt.show()
```

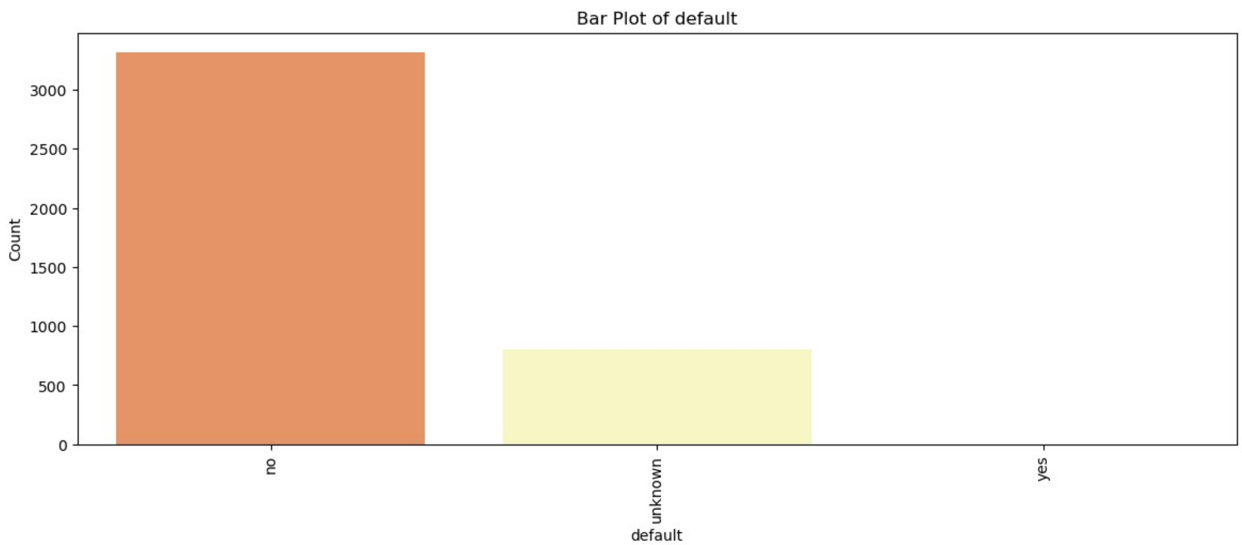


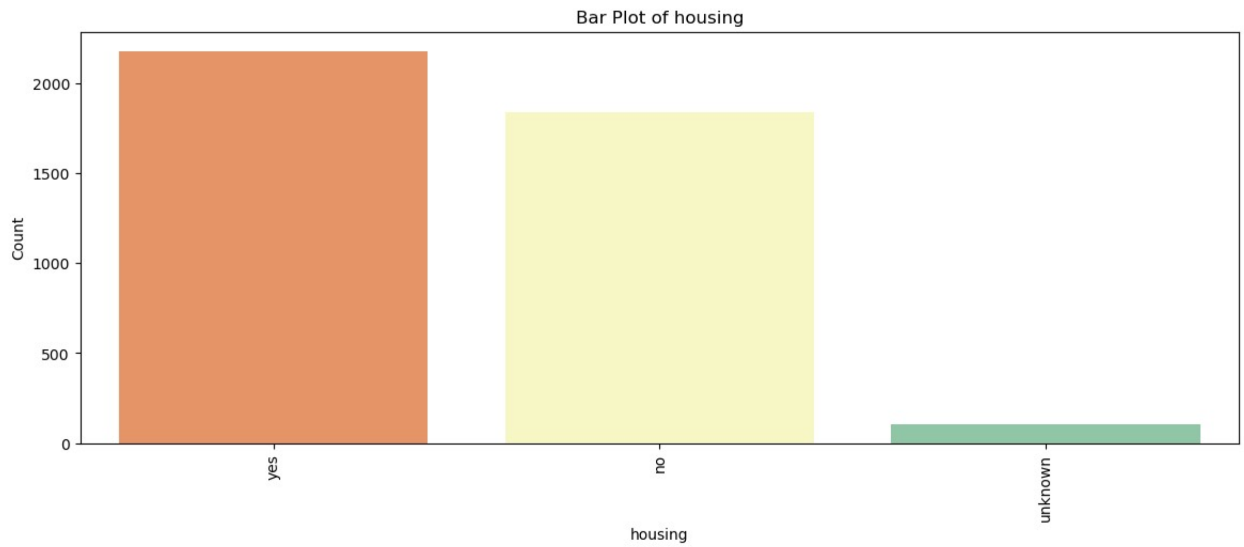

```
for feature in cat_cols:
    plt.figure(figsize=(14,5))  # Adjust the figure size as needed
    sns.countplot(x=feature, data=df, palette='Spectral')
    plt.title(f'Bar Plot of {feature}')
    plt.xlabel(feature)
    plt.ylabel('Count')
    plt.xticks(rotation=90)
    plt.show()
```


Bar Plot of job
Count
0
200
400
600
800
1000
blue-collar
services
admin.
entrepreneur
self-employed
technician
management
student
retired
housemaid
unemployed
unknown
job

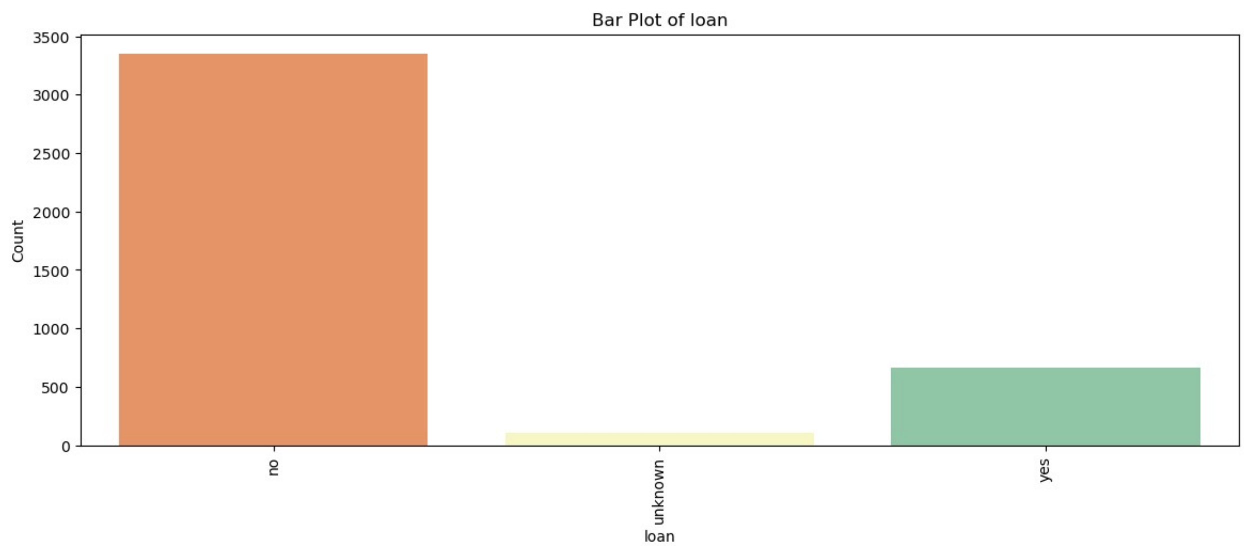

Bar Plot of marital
Count
0
500
1000
1500
2000
2500
married
single
divorced
unknown
marital


Bar Plot of education
Count
0
200
400
600
800
1000
1200
basic.9y
high.school
university.degree
professional.course
basic.6y
basic.4y
unknown
illiterate
education

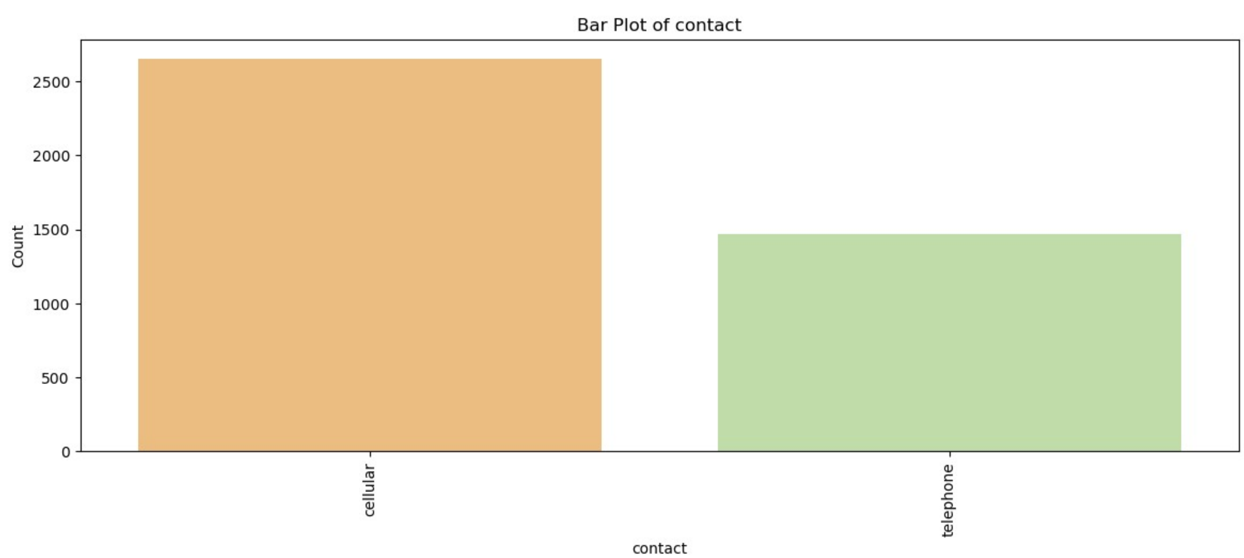

Bar Plot of default
Count
0
500
1000
1500
2000
2500
3000
no
unknown
yes
default



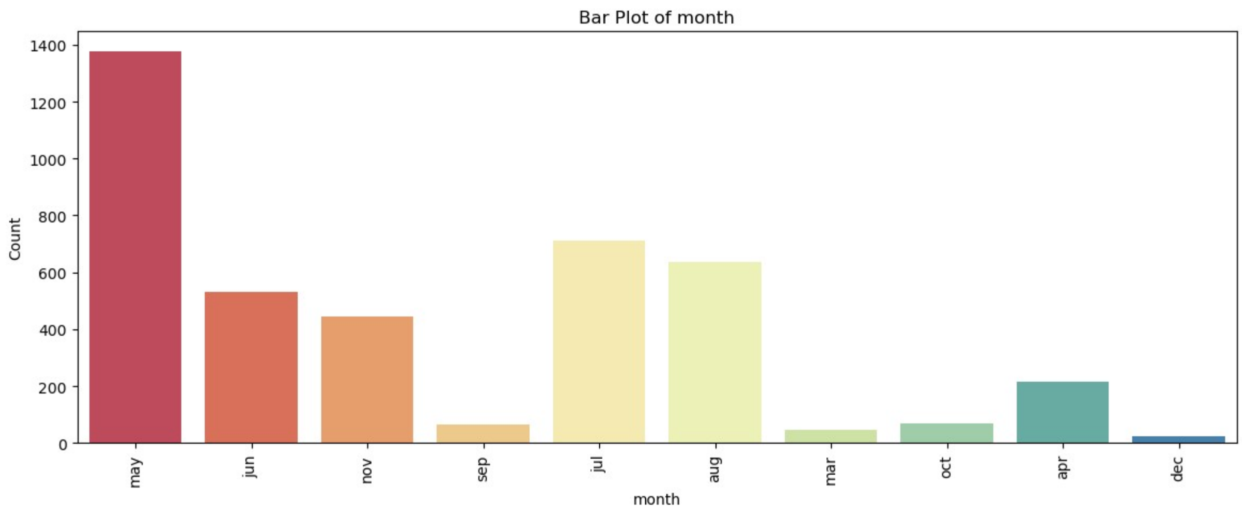
Bar Plot of housing
Count
0
500
1000
1500
2000
yes
no
unknown
housing

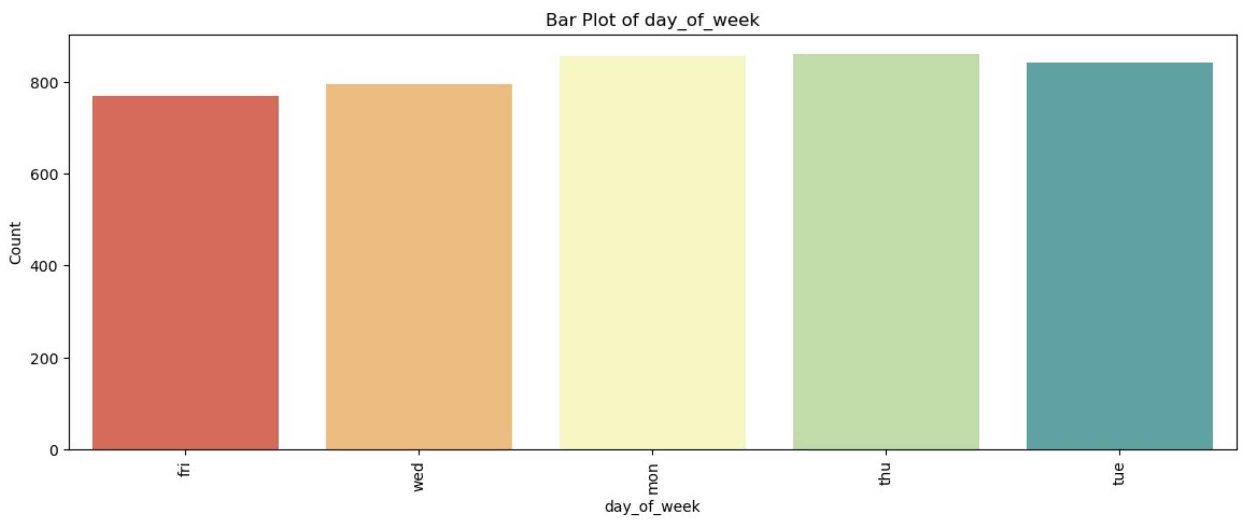


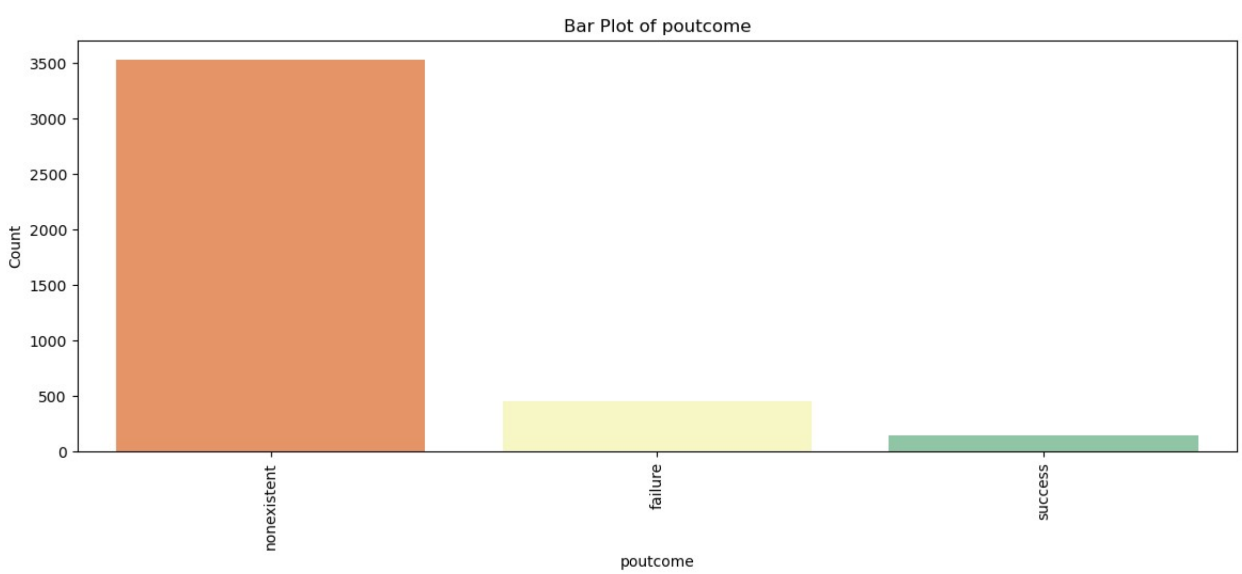
Bar Plot of loan
Count
0
500
1000
1500
2000
2500
3000
3500
no
unknown
yes
loan

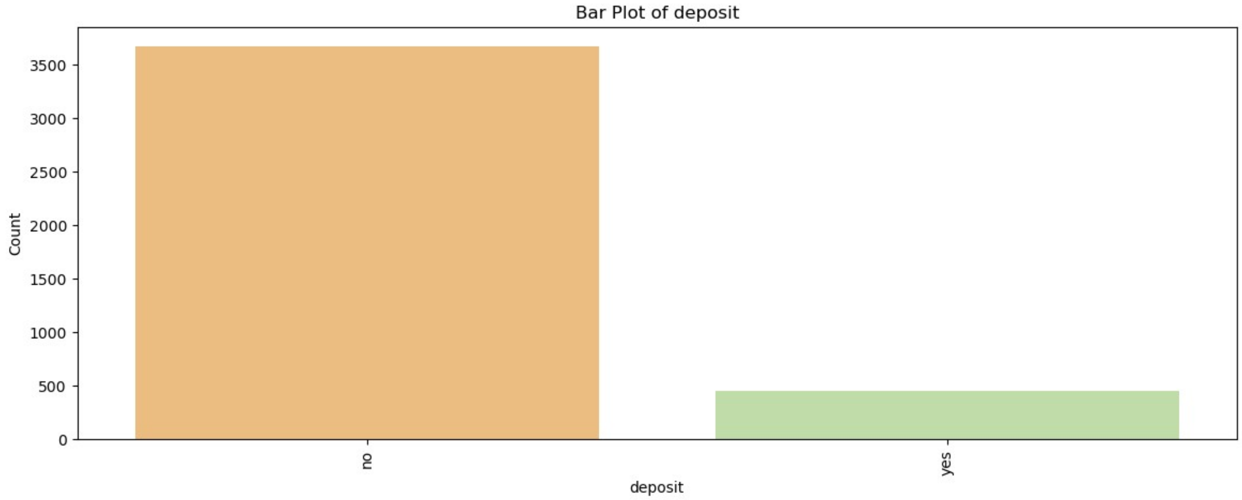


Bar Plot of contact
Count
0
500
1000
1500
2000
2500
cellular
telephone
contact

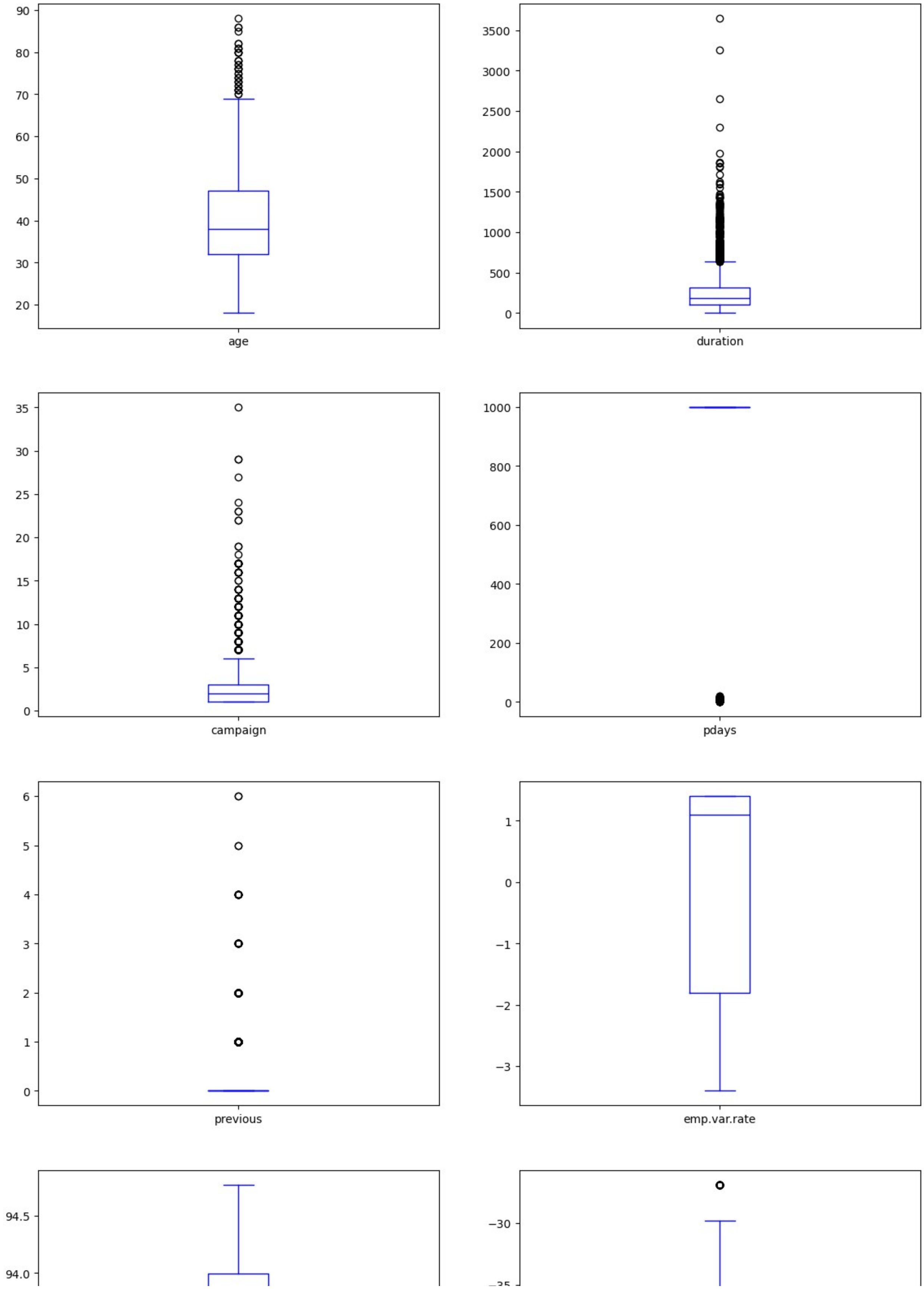

Bar Plot of month
Count
1400
1200
1000
800
600
400
200
0
may
jun
nov
sep
jul
aug
mar
oct
apr
dec
month


Bar Plot of day_of_week
Count
800
600
400
200
0
fri
wed
mon
thu
tue
day_of_week

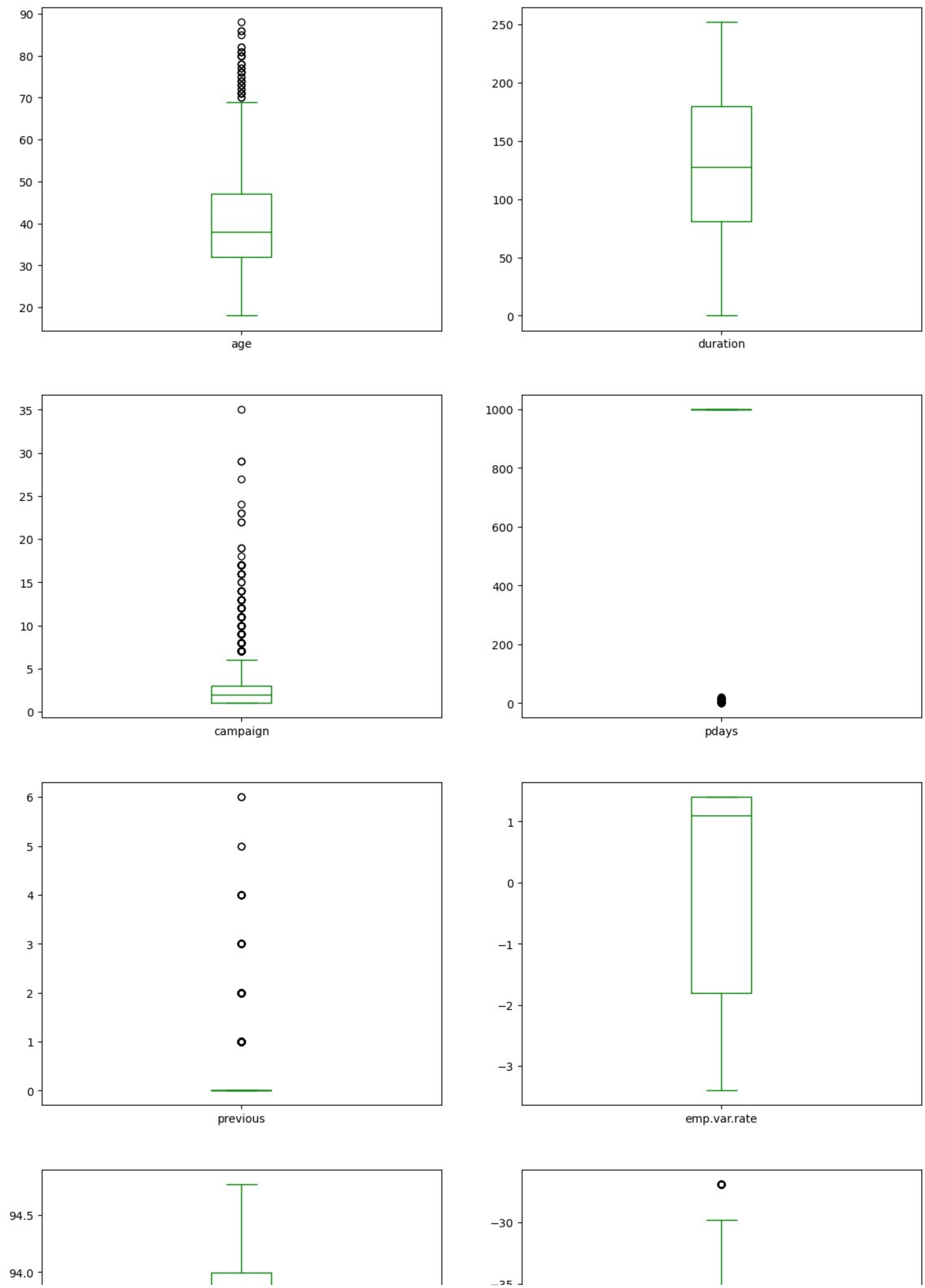

Bar Plot of poutcome
Count
3500
3000
2500
2000
1500
1000
500
0
nonexistent
failure
success
poutcome




```
df.plot(kind='box', subplots=True,
layout=(5,2),figsize=(14,30),color='Blue')
plt.show()
```


90
80
70
60
50
40
30
20
age
3500
3000
2500
2000
1500
1000
500
0
duration
35
30
25
20
15
10
5
0
campaign
1000
800
600
400
200
0
pdays
6
5
4
3
2
1
0
previous
1
0
−1
−2
−3
emp.var.rate
94.5
94.0
−30

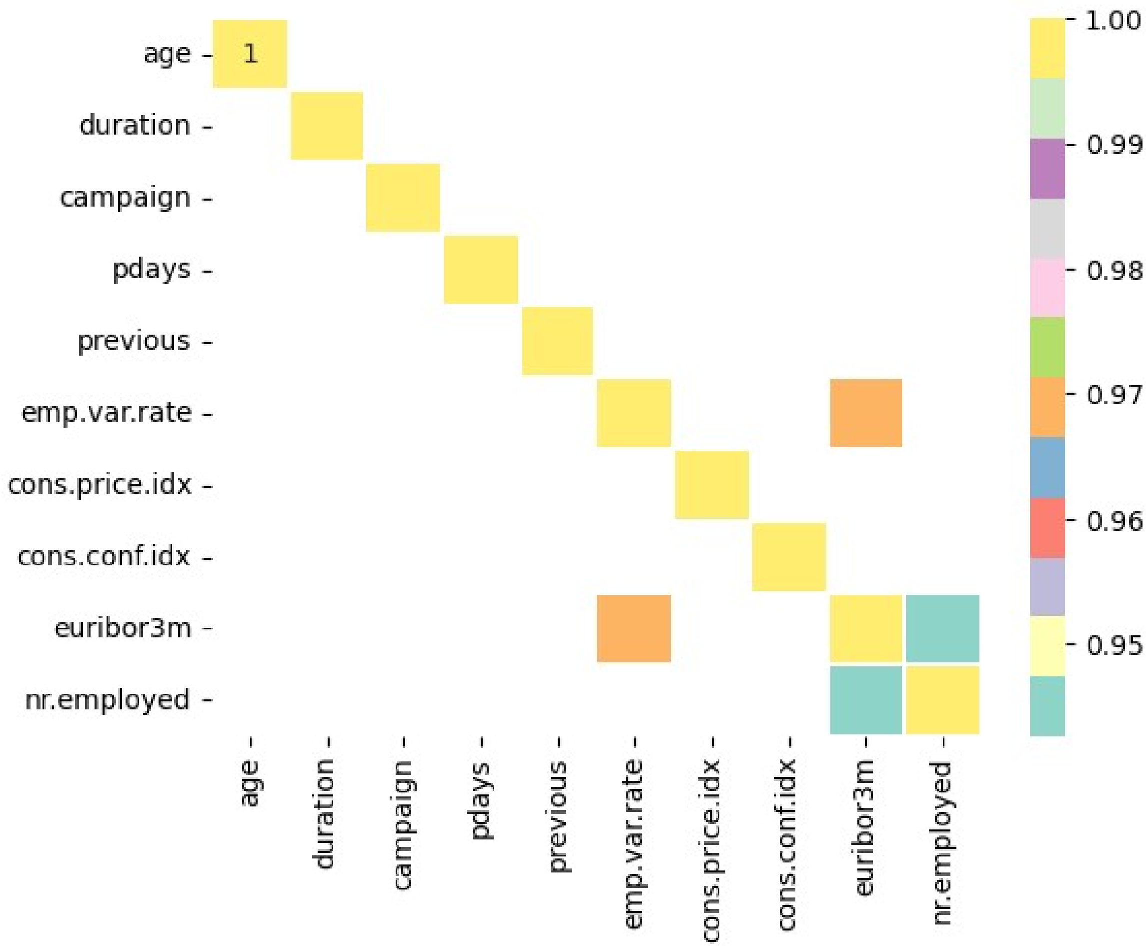
```
column = df[['age','campaign','duration']]
q1 = np.percentile(column, 25)
q3 = np.percentile(column, 75)
iqr = q3 - q1
lower_bound = q1 - 1.5 * iqr
upper_bound = q3 + 1.5 * iqr
df[['age','campaign','duration']] = column[(column > lower_bound) &
(column < upper_bound)]

df.plot(kind='box', subplots=True,
layout=(5,2),figsize=(14,30),color='Green')
plt.show()
```


90
80
70
60
50
40
30
20
age
250
200
150
100
50
0
duration
35
30
25
20
15
10
5
0
campaign
1000
800
600
400
200
0
pdays
6
5
4
3
2
1
0
previous
1
0
−1
−2
−3
emp.var.rate
94.5
94.0
−30

```python
numeric_df = df.select_dtypes(include=[np.number])
corr = numeric_df.corr()
print(corr)
corr = corr[abs(corr) >= 0.90]
sns.heatmap(corr,annot=True,cmap='Set3',linewidths=0.2)
plt.show()
```

```
                     age  duration  campaign     pdays  previous  \
age             1.000000  0.014048 -0.014169 -0.043425  0.050931
duration        0.014048  1.000000 -0.218111 -0.093694  0.094206
campaign       -0.014169 -0.218111  1.000000  0.058742 -0.091490
pdays          -0.043425 -0.093694  0.058742  1.000000 -0.587941
previous        0.050931  0.094206 -0.091490 -0.587941  1.000000
emp.var.rate   -0.019192 -0.063870  0.176079  0.270684 -0.415238
cons.price.idx -0.000482 -0.013338  0.145021  0.058472 -0.164922
cons.conf.idx   0.098135  0.045889  0.007882 -0.092090 -0.051420
euribor3m      -0.015033 -0.067815  0.159435  0.301478 -0.458851
nr.employed    -0.041936 -0.097339  0.161037  0.381983 -0.514853

                emp.var.rate  cons.price.idx  cons.conf.idx  euribor3m
\
age                -0.019192       -0.000482       0.098135  -0.015033

duration           -0.063870       -0.013338       0.045889  -0.067815

campaign            0.176079        0.145021       0.007882   0.159435

pdays               0.270684        0.058472      -0.092090   0.301478

previous           -0.415238       -0.164922      -0.051420  -0.458851

emp.var.rate        1.000000        0.755155       0.195022   0.970308

cons.price.idx      0.755155        1.000000       0.045835   0.657159

cons.conf.idx       0.195022        0.045835       1.000000   0.276595

euribor3m           0.970308        0.657159       0.276595   1.000000

nr.employed         0.897173        0.472560       0.107054   0.942589


                nr.employed
age               -0.041936
duration          -0.097339
campaign           0.161037
pdays              0.381983
previous          -0.514853
emp.var.rate       0.897173
cons.price.idx     0.472560
cons.conf.idx      0.107054
```

```
euribor3m         0.942589
nr.employed       1.000000
```




```
high_corr_cols = ['emp.var.rate','euribor3m','nr.employed']

df1 = df.copy()
df1.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', 'deposit'],
      dtype='object')

df1.shape

(4119, 21)
```

```
from sklearn.preprocessing import LabelEncoder
lb = LabelEncoder()
df_encoded = df1.apply(lb.fit_transform)
df_encoded
```

```
      age  job  marital  education  default  housing  loan  contact  
month  \
0      12    1        1          2        0        2     0        0  
6
1      21    7        2          3        0        0     0        1  
6
2       7    7        1          3        0        2     0        1  
4
3      20    7        1          2        0        1     1        1  
4
4      29    0        1          6        0        2     0        0  
7
...   ...  ...      ...        ...      ...      ...   ...      ...  
...
4114   12    0        1          1        0        2     2        0  
3
4115   21    0        1          3        0        2     0        1  
3
4116    9    8        2          3        0        0     0        0  
6
4117   40    0        1          3        0        0     0        0  
1
4118   16    4        2          3        0        2     0        0  
7

      day_of_week  ...  campaign  pdays  previous  poutcome  
emp.var.rate  \
0               0  ...         1     20         0         1  
3
1               0  ...         3     20         0         1  
8
2               4  ...         0     20         0         1  
9
3               0  ...         2     20         0         1  
9
4               1  ...         0     20         0         1  
7
...           ...  ...       ...    ...       ...       ...  
...
4114            2  ...         0     20         0         1  
9
4115            0  ...         0     20         0         1  
9
4116            1  ...         1     20         1         0  
3
```

```
4117            0  ...         0     20         0         1
9
4118            4  ...         0     20         0         1
7

      cons.price.idx  cons.conf.idx  euribor3m  nr.employed  deposit
0                  8              4        156            6        0
1                 18             16        207            8        0
2                 23              8        225           10        0
3                 23              8        222           10        0
4                 11              7        201            9        0
...              ...            ...        ...          ...      ...
4114              17              6        221           10        0
4115              17              6        222           10        0
4116               8              4        160            6        0
4117              13             17        229           10        0
4118              11              7        199            9        0

[4119 rows x 21 columns]
```

```
df_encoded['deposit'].value_counts()
```

```
deposit
0    3668
1     451
Name: count, dtype: int64
```

```
x = df_encoded.drop('deposit',axis=1)  # independent variable
y = df_encoded['deposit']              # dependent variable
print(x.shape)
print(y.shape)
print(type(x))
print(type(y))
```

```
(4119, 20)
(4119,)
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.series.Series'>
```

```
from sklearn.model_selection import train_test_split
print(4119*0.25)
```

```
1029.75
```

```
x_train,x_test,y_train,y_test =
train_test_split(x,y,test_size=0.25,random_state=1)
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

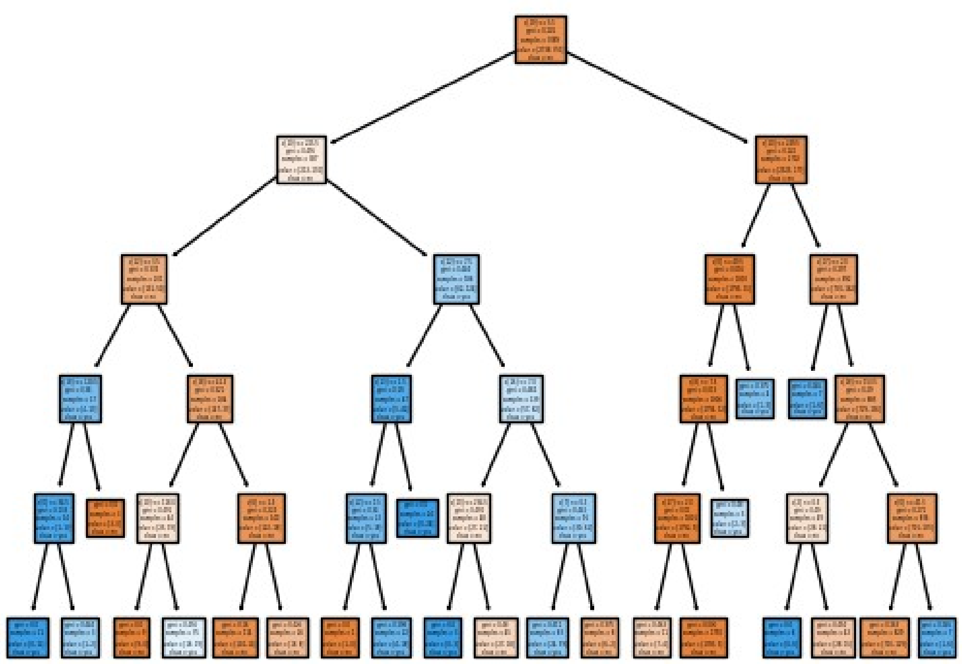
```
(3089, 20)
(1030, 20)
(3089,)
(1030,)

from sklearn.metrics import
confusion_matrix,classification_report,accuracy_score

def eval_model(y_test,y_pred):
    acc = accuracy_score(y_test,y_pred)
    print('Accuracy_Score',acc)
    cm = confusion_matrix(y_test,y_pred)
    print('Confusion Matrix\n',cm)
    print('Classification Report\
n',classification_report(y_test,y_pred))

def mscore(model):
    train_score = model.score(x_train,y_train)
    test_score = model.score(x_test,y_test)
    print('Training Score',train_score)
    print('Testing Score',test_score)

from sklearn.tree import DecisionTreeClassifier

dt =
DecisionTreeClassifier(criterion='gini',max_depth=5,min_samples_split=
10)
dt.fit(x_train,y_train)

DecisionTreeClassifier(max_depth=5, min_samples_split=10)

mscore(dt)

Training Score 0.9219812236969893
Testing Score 0.9087378640776699

ypred_dt = dt.predict(x_test)
print(ypred_dt)

[0 0 1 ... 1 0 0]

eval_model(y_test,ypred_dt)

Accuracy_Score 0.9087378640776699
Confusion Matrix
 [[902  28]
 [ 66  34]]
Classification Report
               precision    recall  f1-score   support

           0       0.93      0.97      0.95       930
           1       0.55      0.34      0.42       100
```

```
    accuracy                           0.91      1030
   macro avg       0.74      0.65      0.69      1030
weighted avg       0.89      0.91      0.90      1030
```

```python
from sklearn.tree import plot_tree

cn = ['no','yes']
fn = x_train.columns
print(fn)
print(cn)
```

```
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'],
      dtype='object')
['no', 'yes']
```

```python
plot_tree(dt,class_names=cn,filled=True)
plt.show()
```



```python
dt1 =
DecisionTreeClassifier(criterion='entropy',max_depth=4,min_samples_spl
```

```
it=15)
dt1.fit(x_train,y_train)
```

```
DecisionTreeClassifier(criterion='entropy', max_depth=4,
min_samples_split=15)
```

```
mscore(dt1)
```

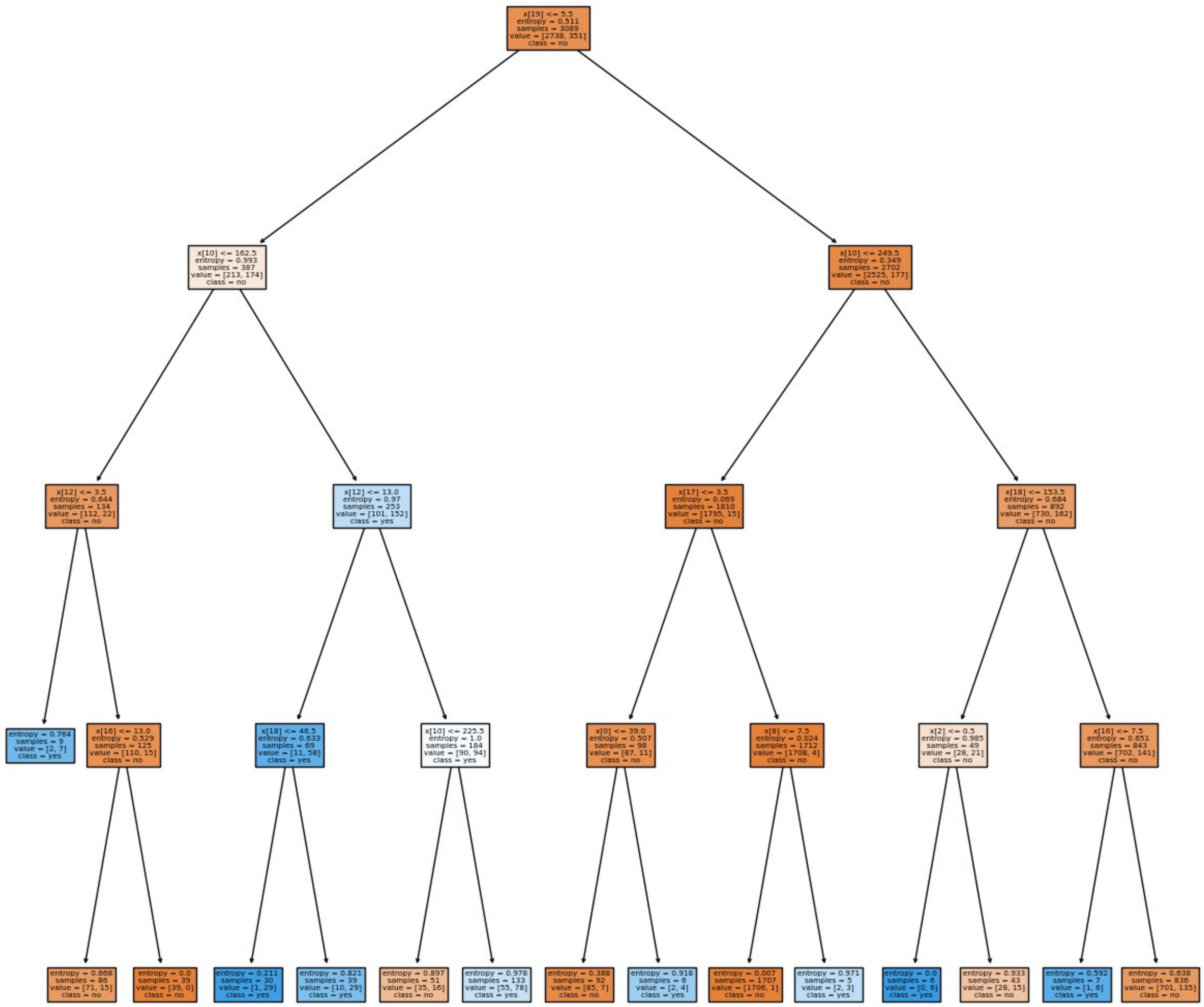
```
Training Score 0.915182907089673
Testing Score 0.9106796116504854
```

```
ypred_dt1 = dt1.predict(x_test)
```

```
eval_model(y_test,ypred_dt1)
```

```
Accuracy_Score 0.9106796116504854
Confusion Matrix
 [[896  34]
 [ 58  42]]
Classification Report
               precision    recall  f1-score   support

           0       0.94      0.96      0.95       930
           1       0.55      0.42      0.48       100

    accuracy                           0.91      1030
   macro avg       0.75      0.69      0.71      1030
weighted avg       0.90      0.91      0.91      1030
```

```
plt.figure(figsize=(15,15))
plot_tree(dt1,class_names=cn,filled=True)
plt.show()
```


x[19] <= 5.5
entropy = 0.511
samples = 3089
value = [2738, 351]
class = no
x[10] <= 162.5
entropy = 0.993
samples = 387
value = [213, 174]
class = no
x[10] <= 249.5
entropy = 0.349
samples = 2702
value = [2525, 177]
class = no
x[12] <= 3.5
entropy = 0.644
samples = 134
value = [112, 22]
class = no
x[12] <= 13.0
entropy = 0.97
samples = 253
value = [101, 152]
class = yes
x[17] <= 3.5
entropy = 0.069
samples = 1810
value = [1795, 15]
class = no
x[18] <= 153.5
entropy = 0.684
samples = 892
value = [730, 162]
class = no
entropy = 0.764
samples = 9
value = [2, 7]
class = yes
x[16] <= 13.0
entropy = 0.529
samples = 125
value = [110, 15]
class = no
x[18] <= 46.5
entropy = 0.633
samples = 69
value = [11, 58]
class = yes
x[10] <= 225.5
entropy = 1.0
samples = 184
value = [90, 94]
class = yes
x[0] <= 39.0
entropy = 0.507
samples = 98
value = [87, 11]
class = no
x[8] <= 7.5
entropy = 0.024
samples = 1712
value = [1708, 4]
class = no
x[2] <= 0.5
entropy = 0.985
samples = 49
value = [28, 21]
class = no
x[16] <= 7.5
entropy = 0.651
samples = 843
value = [702, 141]
class = no
entropy = 0.668
samples = 86
value = [71, 15]
class = no
entropy = 0.0
samples = 39
value = [39, 0]
class = no
entropy = 0.211
samples = 30
value = [1, 29]
class = yes
entropy = 0.821
samples = 39
value = [10, 29]
class = yes
entropy = 0.897
samples = 51
value = [35, 16]
class = no
entropy = 0.978
samples = 133
value = [55, 78]
class = yes
entropy = 0.388
samples = 92
value = [85, 7]
class = no
entropy = 0.918
samples = 6
value = [2, 4]
class = yes
entropy = 0.007
samples = 1707
value = [1706, 1]
class = no
entropy = 0.971
samples = 5
value = [2, 3]
class = yes
entropy = 0.0
samples = 6
value = [0, 6]
class = yes
entropy = 0.933
samples = 43
value = [28, 15]
class = no
entropy = 0.592
samples = 7
value = [1, 6]
class = yes
entropy = 0.638
samples = 836
value = [701, 135]
class = no