ASSIGNMENT_1.2_

August 11, 2024

[31]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.ensemble import AdaBoostClassifier
from sklearn.metrics import␣
 ↪confusion_matrix,accuracy_score,classification_report

import warnings
warnings.filterwarnings('ignore')
```

[32]:
```
sns.get_dataset_names()
```

[32]:
```
['anagrams',
 'anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'dowjones',
 'exercise',
 'flights',
 'fmri',
 'geyser',
 'glue',
 'healthexp',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'seaice',
 'taxis',
 'tips',
 'titanic',
 'anagrams',
```

```
'anagrams',
'anscombe',
'anscombe',
'attention',
'attention',
'brain_networks',
'brain_networks',
'car_crashes',
'car_crashes',
'diamonds',
'diamonds',
'dots',
'dots',
'dowjones',
'dowjones',
'exercise',
'exercise',
'flights',
'flights',
'fmri',
'fmri',
'geyser',
'geyser',
'glue',
'glue',
'healthexp',
'healthexp',
'iris',
'iris',
'mpg',
'mpg',
'penguins',
'penguins',
'planets',
'planets',
'seaice',
'seaice',
'taxis',
'taxis',
'tips',
'tips',
'titanic',
'titanic',
'anagrams',
'anscombe',
'attention',
'brain_networks',
```

```
 'car_crashes',
 'diamonds',
 'dots',
 'dowjones',
 'exercise',
 'flights',
 'fmri',
 'geyser',
 'glue',
 'healthexp',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'seaice',
 'taxis',
 'tips',
 'titanic']
```

[33]:
```
dataset = pd.read_csv(r'D:\loan_detection.csv')
print(dataset)
```

```
       age  campaign  pdays  previous  no_previous_contact  not_working  \
0       56         1    999         0                    1            0
1       57         1    999         0                    1            0
2       37         1    999         0                    1            0
3       40         1    999         0                    1            0
4       56         1    999         0                    1            0
…      …        …     …        …                    …            …
41183   73         1    999         0                    1            1
41184   46         1    999         0                    1            0
41185   56         2    999         0                    1            1
41186   44         1    999         0                    1            0
41187   74         3    999         1                    1            1

       job_admin.  job_blue-collar  job_entrepreneur  job_housemaid  …  \
0               0                0                 0              1  …
1               0                0                 0              0  …
2               0                0                 0              0  …
3               1                0                 0              0  …
4               0                0                 0              0  …
…             …              …                 …            …  …
41183           0                0                 0              0  …
41184           0                1                 0              0  …
41185           0                0                 0              0  …
41186           0                0                 0              0  …
41187           0                0                 0              0  …
```

```
       month_sep  day_of_week_fri  day_of_week_mon  day_of_week_thu  \
0              0                0                1                0
1              0                0                1                0
2              0                0                1                0
3              0                0                1                0
4              0                0                1                0
…            …                …                …                …
41183          0                1                0                0
41184          0                1                0                0
41185          0                1                0                0
41186          0                1                0                0
41187          0                1                0                0

       day_of_week_tue  day_of_week_wed  poutcome_failure  \
0                    0                0                 0
1                    0                0                 0
2                    0                0                 0
3                    0                0                 0
4                    0                0                 0
…                  …                …                 …
41183                0                0                 0
41184                0                0                 0
41185                0                0                 0
41186                0                0                 0
41187                0                0                 1

       poutcome_nonexistent  poutcome_success  Loan_Status_label
0                         1                 0                  0
1                         1                 0                  0
2                         1                 0                  0
3                         1                 0                  0
4                         1                 0                  0
…                       …                 …                  …
41183                     1                 0                  1
41184                     1                 0                  0
41185                     1                 0                  0
41186                     1                 0                  1
41187                     0                 0                  0

[41188 rows x 60 columns]
```

[34]:
```
dataset.columns
```

[34]:
```
Index(['age', 'campaign', 'pdays', 'previous', 'no_previous_contact',
       'not_working', 'job_admin.', 'job_blue-collar', 'job_entrepreneur',
       'job_housemaid', 'job_management', 'job_retired', 'job_self-employed',
       'job_services', 'job_student', 'job_technician', 'job_unemployed',
```

```
       'job_unknown', 'marital_divorced', 'marital_married', 'marital_single',
       'marital_unknown', 'education_basic.4y', 'education_basic.6y',
       'education_basic.9y', 'education_high.school', 'education_illiterate',
       'education_professional.course', 'education_university.degree',
       'education_unknown', 'default_no', 'default_unknown', 'default_yes',
       'housing_no', 'housing_unknown', 'housing_yes', 'loan_no',
       'loan_unknown', 'loan_yes', 'contact_cellular', 'contact_telephone',
       'month_apr', 'month_aug', 'month_dec', 'month_jul', 'month_jun',
       'month_mar', 'month_may', 'month_nov', 'month_oct', 'month_sep',
       'day_of_week_fri', 'day_of_week_mon', 'day_of_week_thu',
       'day_of_week_tue', 'day_of_week_wed', 'poutcome_failure',
       'poutcome_nonexistent', 'poutcome_success', 'Loan_Status_label'],
      dtype='object')
```

[35]:
```
X=dataset[['age','job_housemaid','job_management','marital_married','contact_cellular']]
X
```

[35]:

	age	job_housemaid	job_management	marital_married	contact_cellular
0	56	1	0	1	0
1	57	0	0	1	0
2	37	0	0	1	0
3	40	0	0	1	0
4	56	0	0	1	0
…	…	…	…	…	…
41183	73	0	0	1	1
41184	46	0	0	1	1
41185	56	0	0	1	1
41186	44	0	0	1	1
41187	74	0	0	1	1

[41188 rows x 5 columns]

[36]:
```
y=dataset['education_illiterate']
y
```

[36]:
```
0        0
1        0
2        0
3        0
4        0
        ..
41183    0
41184    0
41185    0
41186    0
41187    0
Name: education_illiterate, Length: 41188, dtype: int64
```

[37]:
```
dataset['education_illiterate'].value_counts()
```

[37]:
```
education_illiterate
0    41170
1       18
Name: count, dtype: int64
```

[38]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.
↪2,random_state=42)
```

[39]:
```
X_train
```

[39]:

	age	job_housemaid	job_management	marital_married	contact_cellular
12556	40	0	0	1	0
35451	31	0	0	1	1
30592	59	0	0	1	1
17914	43	1	0	0	1
3315	39	0	0	0	0
…	…	…	…	…	…
6265	58	0	0	1	0
11284	37	0	1	1	0
38158	35	0	0	1	1
860	40	0	1	1	0
15795	29	0	0	0	1

[32950 rows x 5 columns]

[40]:
```
X_test
```

[40]:

	age	job_housemaid	job_management	marital_married	contact_cellular
32884	57	0	0	1	1
3169	55	0	0	1	0
32206	33	0	0	1	1
9403	36	0	0	1	0
14020	27	1	0	1	1
…	…	…	…	…	…
12322	27	0	0	1	0
23440	41	0	0	0	1
29431	46	0	0	0	1
16627	31	0	0	0	1
1871	59	0	0	1	0

[8238 rows x 5 columns]

[39]:
```
#Decision Tree Pre-Pruning
```

[41]:
```
parameters={
    'criterion':['gini','entropy','log_loss'],
    'splitter' :['best','random'],
    'max_depth':[1,2,3,4,5],
    'max_features':['auto','sqrt','log2'],
    }
```

[42]:
```
from sklearn.model_selection import GridSearchCV
```

[43]:
```
from sklearn.tree import DecisionTreeClassifier
```

[44]:
```
dtree=DecisionTreeClassifier(random_state=42)
grid=GridSearchCV(dtree,param_grid=parameters,cv=5,scoring='accuracy')
grid.fit(X_train,y_train)
```

[44]:
```
GridSearchCV(cv=5, estimator=DecisionTreeClassifier(random_state=42),
             param_grid={'criterion': ['gini', 'entropy', 'log_loss'],
                         'max_depth': [1, 2, 3, 4, 5],
                         'max_features': ['auto', 'sqrt', 'log2'],
                         'splitter': ['best', 'random']},
             scoring='accuracy')
```

[45]:
```
grid.best_params_
```

[45]:
```
{'criterion': 'gini',
 'max_depth': 1,
 'max_features': 'auto',
 'splitter': 'best'}
```

[46]:
```
y_train_pred=grid.predict(X_train)
y_test_pred=grid.predict(X_test)
```

[47]:
```
confusion_matrix(y_train,y_train_pred)
```

[47]:
```
array([[32937,     0],
       [   13,     0]], dtype=int64)
```

[48]:
```
accuracy_score(y_train,y_train_pred)
```

[48]:
```
0.9996054628224583
```

[49]:
```
print(classification_report(y_train,y_train_pred))
```

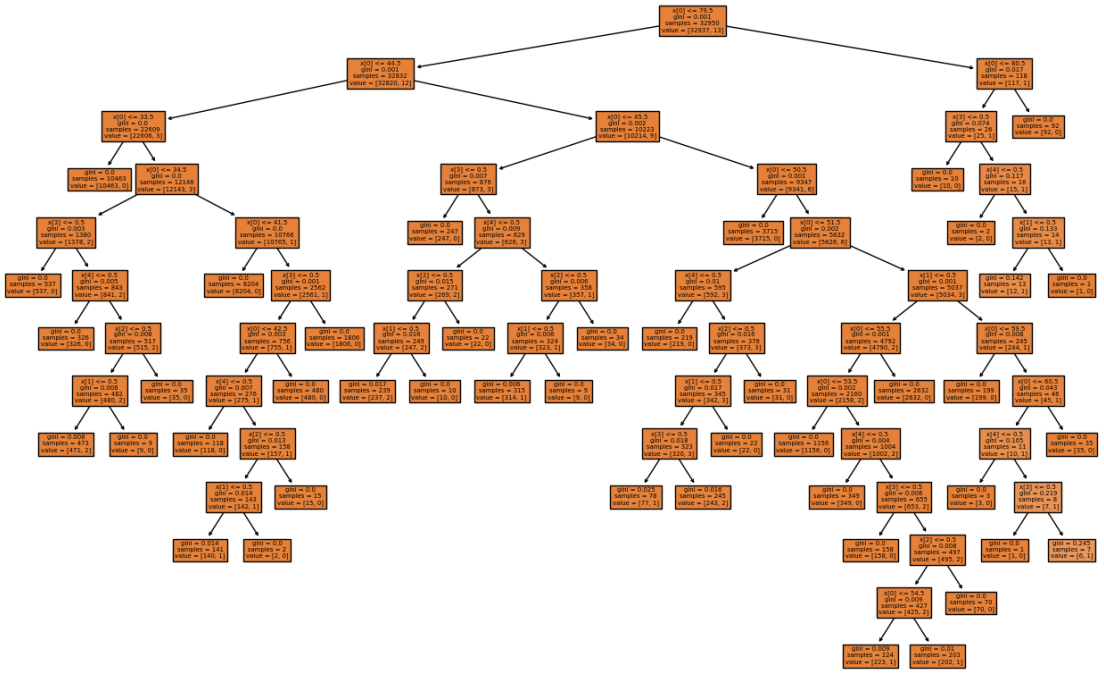
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     32937
           1       0.00      0.00      0.00        13

    accuracy                           1.00     32950
```

```
   macro avg       0.50      0.50      0.50     32950
weighted avg       1.00      1.00      1.00     32950
```

[50]:
```
dtree=DecisionTreeClassifier()
dtree.fit(X_train,y_train)
```

[50]:
```
DecisionTreeClassifier()
```

[51]:
```
from sklearn import tree
plt.figure(figsize=(16,10))
tree.plot_tree(dtree,filled=True)
plt.show()
```



[52]:
```
y_train_pred=grid.predict(X_train)
y_test_pred=grid.predict(X_test)
```

[53]:
```
confusion_matrix(y_train,y_train_pred)
```

[53]:
```
array([[32937,     0],
       [   13,     0]], dtype=int64)
```

[54]:
```
accuracy_score(y_train,y_train_pred)
```

[54]:
```
0.9996054628224583
```

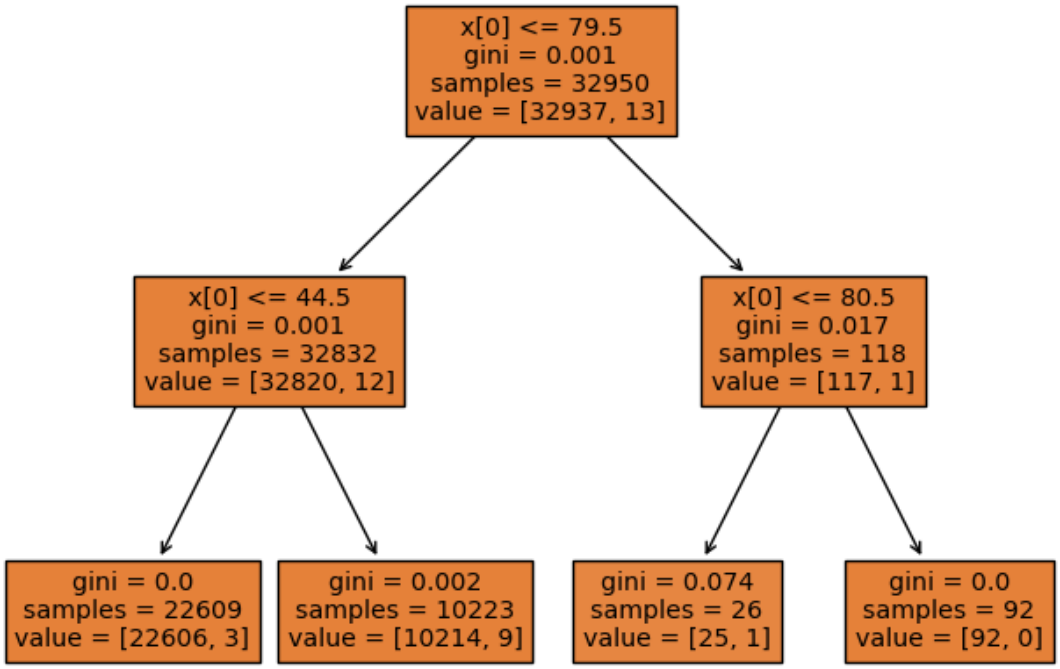
[55]:
```
print(classification_report(y_train,y_train_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     32937
           1       0.00      0.00      0.00        13

    accuracy                           1.00     32950
   macro avg       0.50      0.50      0.50     32950
weighted avg       1.00      1.00      1.00     32950
```

[56]:
```
dtree=DecisionTreeClassifier(max_depth=2)
dtree.fit(X_train,y_train)
```

[56]:
```
DecisionTreeClassifier(max_depth=2)
```

[60]:
```
plt.figure(figsize=(8,6))
tree.plot_tree(dtree,filled=True)
plt.show()
```



[57]:
```
y_train_pred=grid.predict(X_train)
y_test_pred=grid.predict(X_test)
```

[58]:
```
confusion_matrix(y_train,y_train_pred)
```

[58]:
```
array([[32937,     0],
       [   13,     0]], dtype=int64)
```

[61]:
```
accuracy_score(y_train,y_train_pred)
```

[61]:
```
0.9996054628224583
```

[62]:
```
print(classification_report(y_train,y_train_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     32937
           1       0.00      0.00      0.00        13

    accuracy                           1.00     32950
   macro avg       0.50      0.50      0.50     32950
weighted avg       1.00      1.00      1.00     32950
```

[76]:
```
#Logistic Regression
```

[63]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
```

[64]:
```
lr=LogisticRegression()
lr.fit(X_train,y_train)
lr_pred_train=lr.predict(X_train)
lr_pred_test=lr.predict(X_test)
```

[65]:
```
print(confusion_matrix(y_train,lr_pred_train))
print()
print(accuracy_score(y_train,lr_pred_train))
```

```
[[32937     0]
 [   13     0]]

0.9996054628224583
```

[66]:
```
print(confusion_matrix(y_test,lr_pred_test))
print()
print(accuracy_score(y_test,lr_pred_test))
```

```
[[8233    0]
 [   5    0]]

0.999393056567128
```

[78]:
```
#RandomForest
```

[67]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

[68]:
```
rf=RandomForestClassifier()
rf.fit(X_train,y_train)
rf_pred_train=rf.predict(X_train)
rf_pred_test=rf.predict(X_test)
```

[69]:
```
print(confusion_matrix(y_train,rf_pred_train))
print()
print(accuracy_score(y_train,rf_pred_train))
```

```
[[32937     0]
 [   13     0]]

0.9996054628224583
```

[70]:
```
print(confusion_matrix(y_test,rf_pred_test))
print()
print(accuracy_score(y_test,rf_pred_test))
```

```
[[8233    0]
 [   5    0]]

0.999393056567128
```

[92]:
```
#oob_score
```

[71]:
```
rf=RandomForestClassifier(oob_score=True,random_state=42)
rf.fit(X_train,y_train)
rf_pred_train=rf.predict(X_train)
rf_pred_test=rf.predict(X_test)
```

[72]:
```
rf.oob_score_
```

[72]:
```
0.9996054628224583
```

[73]:
```
y_pred_test=rf.predict(X_test)
```

[74]:
```
accuracy_score(y_test,y_pred_test)
```

[74]:
```
0.999393056567128
```

[75]:
```
X_train.shape,X_test.shape
```

[75]:
```
((32950, 5), (8238, 5))
```

[76]:
```
y_train.shape,y_test.shape
```

[76]:
```
((32950,), (8238,))
```

[12]:
```
#Slopes or Coefficent
```

[77]:
```
X.columns
```

[77]:
```
Index(['age', 'job_housemaid', 'job_management', 'marital_married',
       'contact_cellular'],
      dtype='object')
```

[16]:
```
lr.coef_
```

[16]:
```
array([[ 0.05975469,  0.30722461, -0.58259686,  0.66040151,  0.71055723]])
```

[78]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import␣
 ↪confusion_matrix,accuracy_score,classification_report
from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
import warnings
warnings.filterwarnings('ignore')
```

[79]:
```
dataset.nunique()
```

[79]:
```
age                    78
campaign               42
pdays                  27
previous                8
no_previous_contact     2
not_working             2
job_admin.              2
job_blue-collar         2
job_entrepreneur        2
job_housemaid           2
```

job_management	2
job_retired	2
job_self-employed	2
job_services	2
job_student	2
job_technician	2
job_unemployed	2
job_unknown	2
marital_divorced	2
marital_married	2
marital_single	2
marital_unknown	2
education_basic.4y	2
education_basic.6y	2
education_basic.9y	2
education_high.school	2
education_illiterate	2
education_professional.course	2
education_university.degree	2
education_unknown	2
default_no	2
default_unknown	2
default_yes	2
housing_no	2
housing_unknown	2
housing_yes	2
loan_no	2
loan_unknown	2
loan_yes	2
contact_cellular	2
contact_telephone	2
month_apr	2
month_aug	2
month_dec	2
month_jul	2
month_jun	2
month_mar	2
month_may	2
month_nov	2
month_oct	2
month_sep	2
day_of_week_fri	2
day_of_week_mon	2
day_of_week_thu	2
day_of_week_tue	2
day_of_week_wed	2
poutcome_failure	2

```
poutcome_nonexistent             2
poutcome_success                 2
Loan_Status_label                2
dtype: int64
```

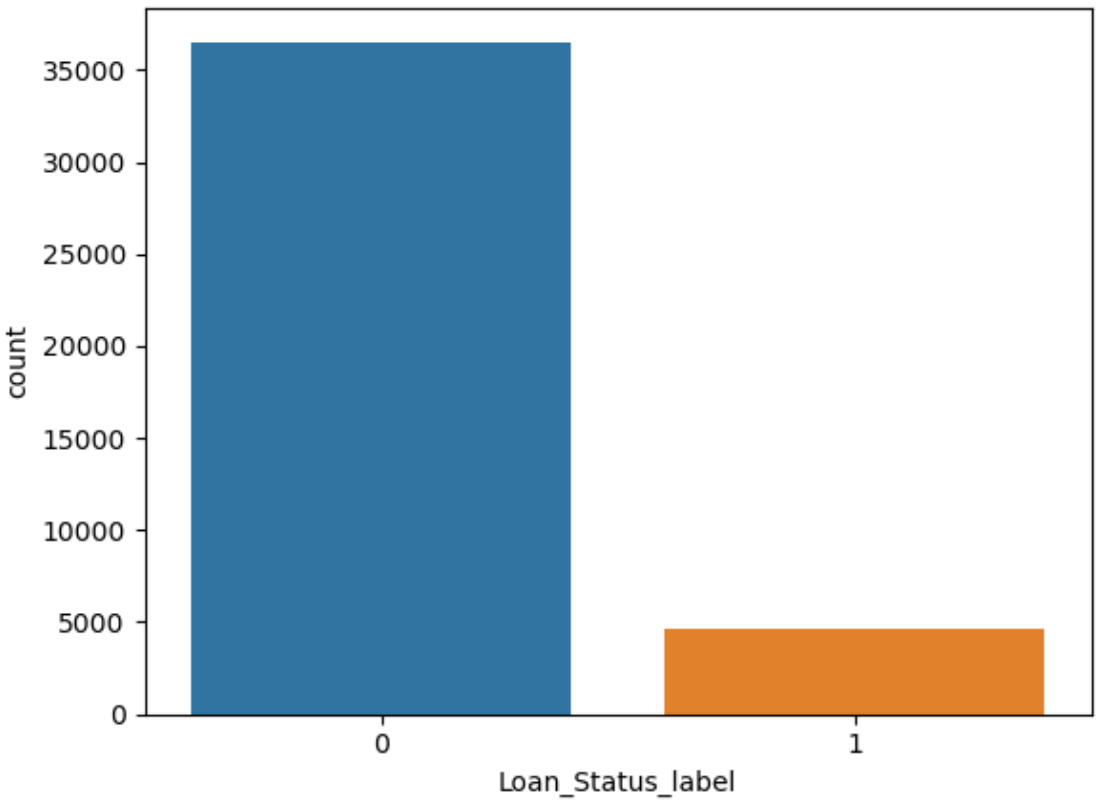
[80]:
```
dataset.isnull().sum()
```

[80]:
```
age                              0
campaign                         0
pdays                            0
previous                         0
no_previous_contact              0
not_working                      0
job_admin.                       0
job_blue-collar                  0
job_entrepreneur                 0
job_housemaid                    0
job_management                   0
job_retired                      0
job_self-employed                0
job_services                     0
job_student                      0
job_technician                   0
job_unemployed                   0
job_unknown                      0
marital_divorced                 0
marital_married                  0
marital_single                   0
marital_unknown                  0
education_basic.4y               0
education_basic.6y               0
education_basic.9y               0
education_high.school            0
education_illiterate             0
education_professional.course    0
education_university.degree      0
education_unknown                0
default_no                       0
default_unknown                  0
default_yes                      0
housing_no                       0
housing_unknown                  0
housing_yes                      0
loan_no                          0
loan_unknown                     0
loan_yes                         0
contact_cellular                 0
```

```
contact_telephone               0
month_apr                       0
month_aug                       0
month_dec                       0
month_jul                       0
month_jun                       0
month_mar                       0
month_may                       0
month_nov                       0
month_oct                       0
month_sep                       0
day_of_week_fri                 0
day_of_week_mon                 0
day_of_week_thu                 0
day_of_week_tue                 0
day_of_week_wed                 0
poutcome_failure                0
poutcome_nonexistent            0
poutcome_success                0
Loan_Status_label               0
dtype: int64
```

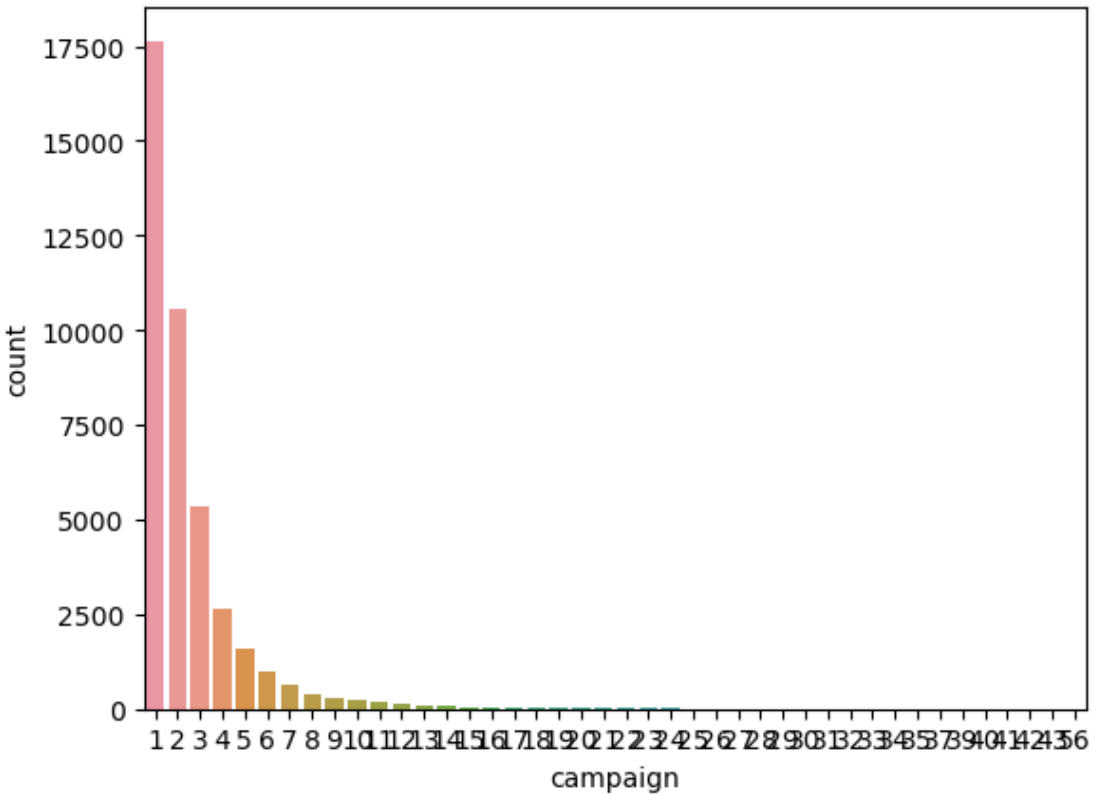
[82]:
```
sns.countplot(x='Loan_Status_label',data=dataset)
```

[82]:
```
<Axes: xlabel='Loan_Status_label', ylabel='count'>
```



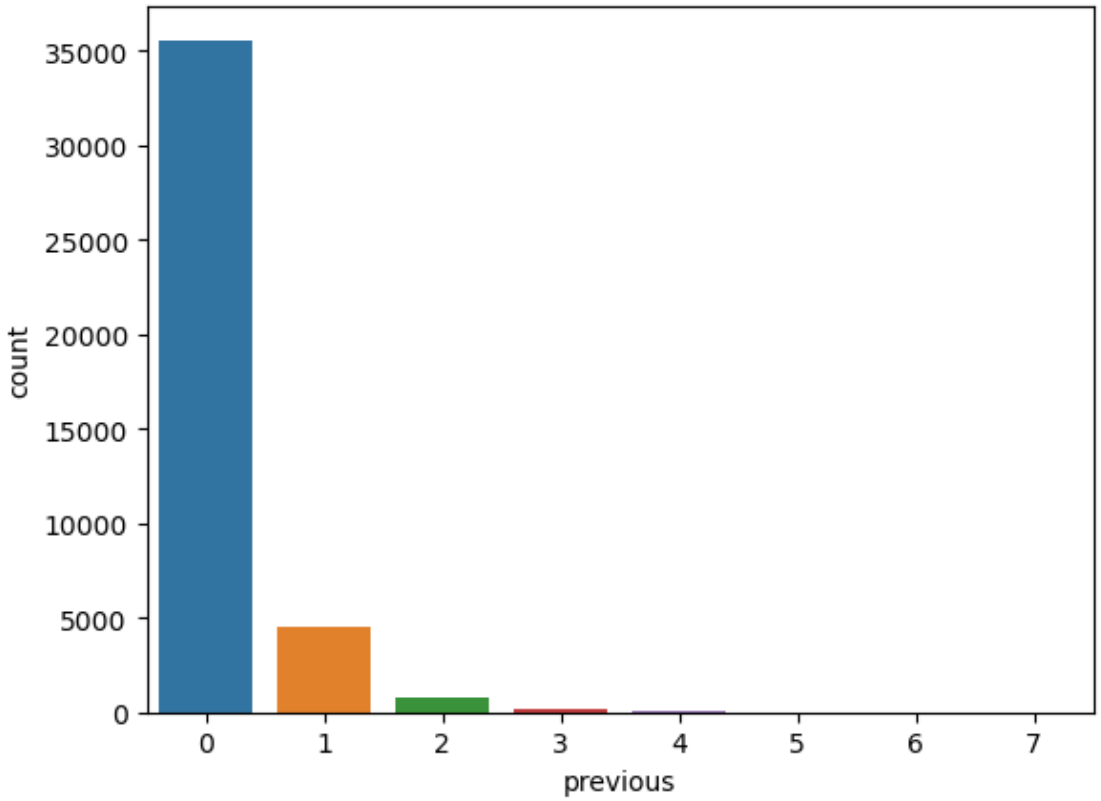
[83]:
```
sns.countplot(x='campaign',data=dataset)
```

[83]:
```
<Axes: xlabel='campaign', ylabel='count'>
```

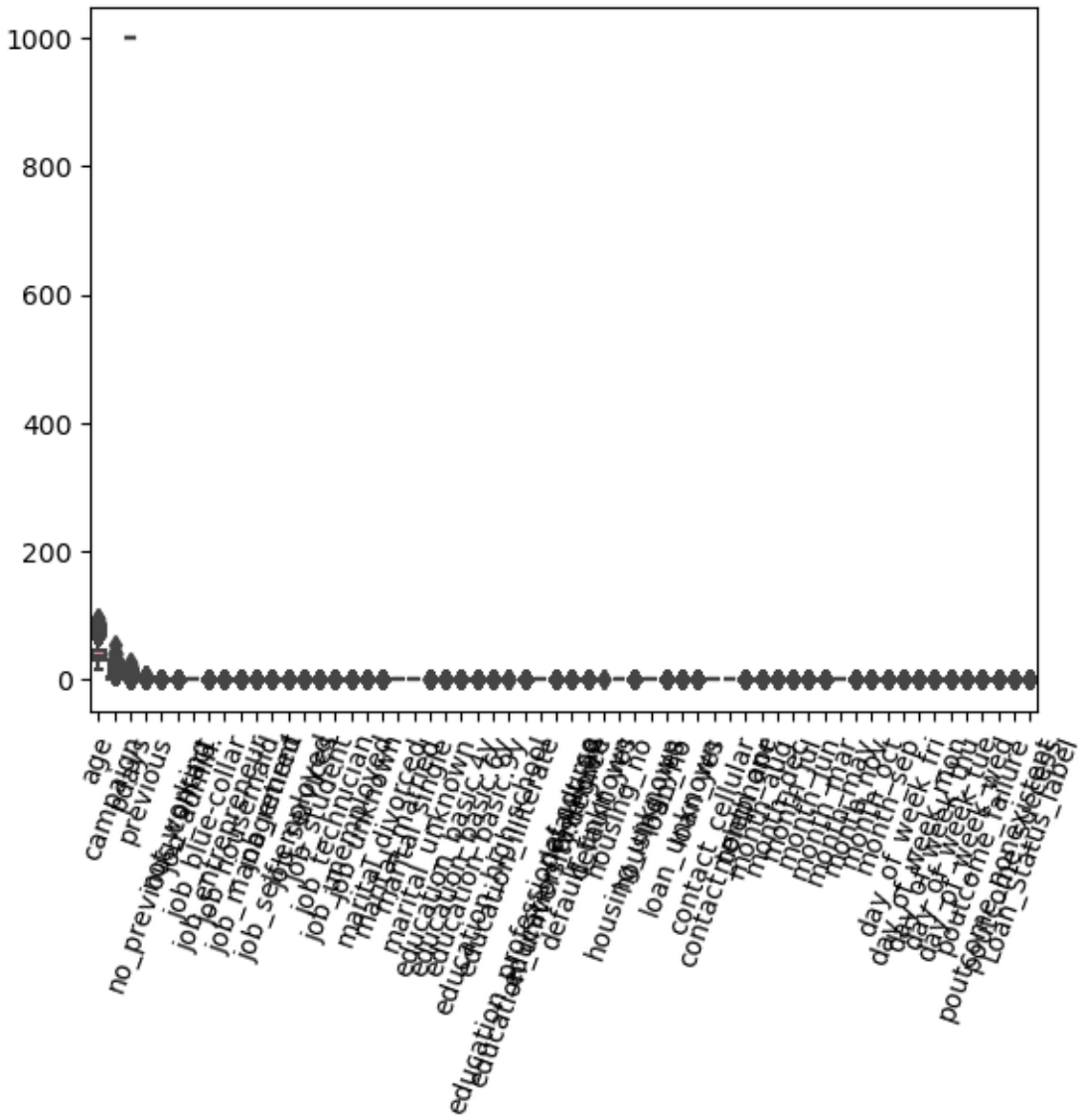


[84]:
```
sns.countplot(x='previous',data=dataset)
```

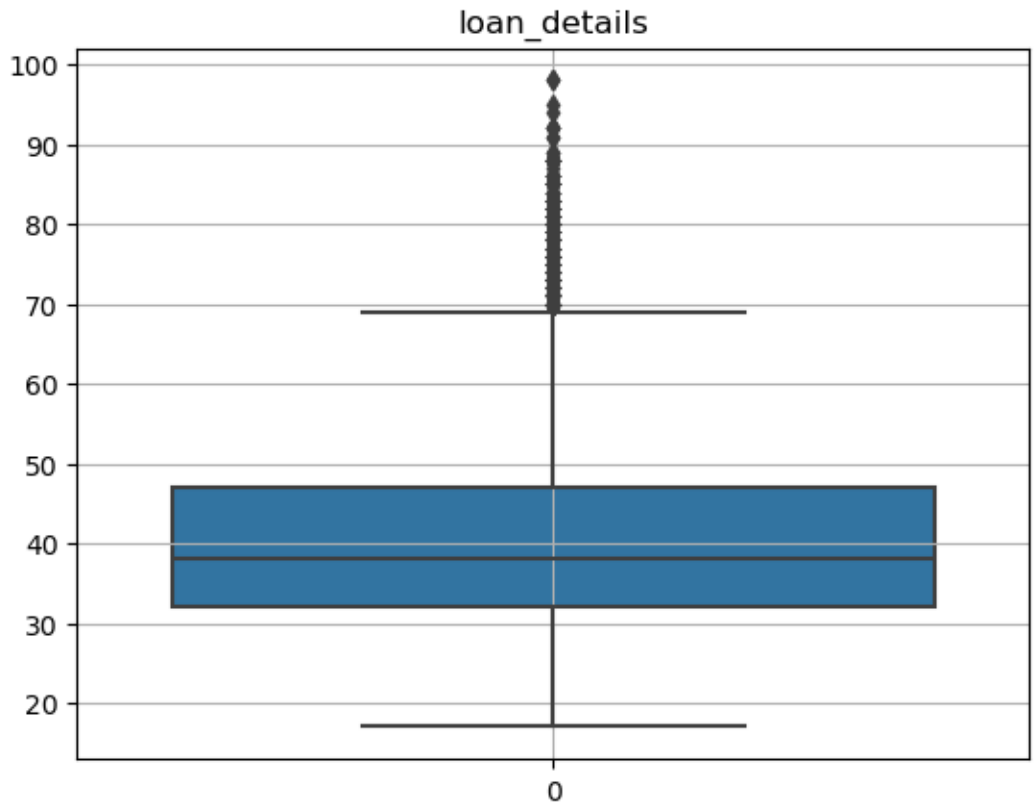
[84]:
```
<Axes: xlabel='previous', ylabel='count'>
```



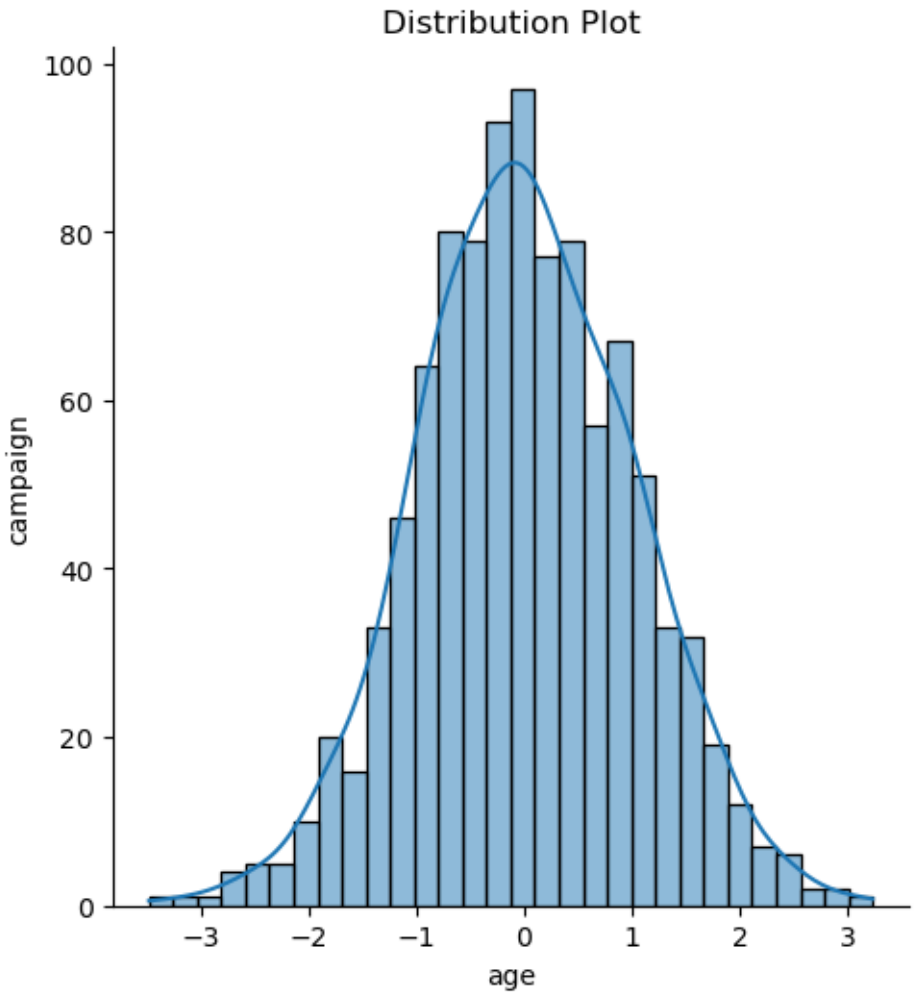
[87]:
```
sns.boxplot(dataset)
plt.xticks(rotation=70)
plt.show()
```



[88]:
```
sns.boxplot(dataset['age'])
plt.title('loan_details')
plt.grid()
plt.show()
```



[92]:
```
# Import libraries
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# Generate random data
data = np.random.normal(loc=0, scale=1, size=1000)  # mean=0, std=1, n=1000

# Create a distribution plot
sns.displot(data, kde=True, bins=30)  # kde=True adds a kernel density estimate
plt.title('Distribution Plot')
plt.xlabel('age')
plt.ylabel('campaign')
plt.show()
```


Distribution Plot
100
80
60
40
20
0
campaign
−3
−2
−1
0
1
2
3
age

[ ]: