Import Libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import metrics

pd.set_option('Display.max_columns',None)
```

In [2]:
```
import warnings
warnings.filterwarnings('ignore')
```

Inspection of data

In [3]:
```
loan_data=pd.read_excel('Copy of case_study_data Max life Ins.xlsx')
```

In [4]:
```
loan_data
```

Out[4]:

	checkin_acc	duration	credit_history	purpose	amount	svaing_acc	present_emp_
0	A11	9	A34	A43	1754	A65	
1	A12	72	A32	A43	8927	A61	
2	A14	18	A34	A46	3144	A61	
3	A11	63	A32	A42	11823	A61	
4	A11	36	A33	A40	7305	A61	
...	...	...	...	...	...	...	
995	A14	18	A32	A42	2604	A61	
996	A11	45	A32	A41	5786	A61	
997	A14	18	A32	A43	1206	A61	
998	A11	68	A32	A43	2768	A61	
999	A12	68	A34	A41	6864	A62	

1000 rows × 21 columns

In [5]:
```
loan_data.shape
```

Out[5]:
```
(1000, 21)
```

In [6]:
```
#It will give values for numerical columns only
loan_data.describe()
```

Out[6]:

	duration	amount	inst_rate	residing_since	age	num_cred
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.0000
mean	31.449000	4907.134000	4.606000	4.407000	53.566000	2.4410
std	18.055564	4234.100176	1.572016	1.570292	17.065664	0.6978
min	6.000000	375.000000	2.000000	2.000000	29.000000	2.0000
25%	18.000000	2048.250000	3.000000	3.000000	41.000000	2.0000
50%	27.000000	3479.500000	5.000000	5.000000	50.000000	2.0000
75%	36.000000	5958.500000	6.000000	6.000000	63.000000	3.0000
max	108.000000	27636.000000	6.000000	6.000000	113.000000	6.0000

In [7]:
```
loan_data.info(verbose=True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 21 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   checkin_acc         1000 non-null   object
 1   duration            1000 non-null   int64 
 2   credit_history      1000 non-null   object
 3   purpose             1000 non-null   object
 4   amount              1000 non-null   int64 
 5   svaing_acc          1000 non-null   object
 6   present_emp_since   1000 non-null   object
 7   inst_rate           1000 non-null   int64 
 8   personal_status     1000 non-null   object
 9   other_debtors       1000 non-null   object
 10  residing_since      1000 non-null   int64 
 11  property            1000 non-null   object
 12  age                 1000 non-null   int64 
 13  inst_plans          1000 non-null   object
 14  housing             1000 non-null   object
 15  num_credits         1000 non-null   int64 
 16  job                 1000 non-null   object
 17  dependents          1000 non-null   int64 
 18  telephone           1000 non-null   object
 19  foreign_worker      1000 non-null   object
 20  status              1000 non-null   int64 
dtypes: int64(8), object(13)
memory usage: 164.2+ KB
```

In [8]:
```
loan_data.head()
```

Out[8]:

	checkin_acc	duration	credit_history	purpose	amount	svaing_acc	present_emp_sir
0	A11	9	A34	A43	1754	A65	A
1	A12	72	A32	A43	8927	A61	A
2	A14	18	A34	A46	3144	A61	A
3	A11	63	A32	A42	11823	A61	A
4	A11	36	A33	A40	7305	A61	A

Checking for missing values

In [9]:
```
#Here we get the number of null values in the data
loan_data.isnull().sum()
```

Out[9]:
```
checkin_acc          0
duration             0
credit_history       0
purpose              0
amount               0
svaing_acc           0
present_emp_since    0
inst_rate            0
personal_status      0
other_debtors        0
residing_since       0
property             0
age                  0
inst_plans           0
housing              0
num_credits          0
job                  0
dependents           0
telephone            0
foreign_worker       0
status               0
dtype: int64
```

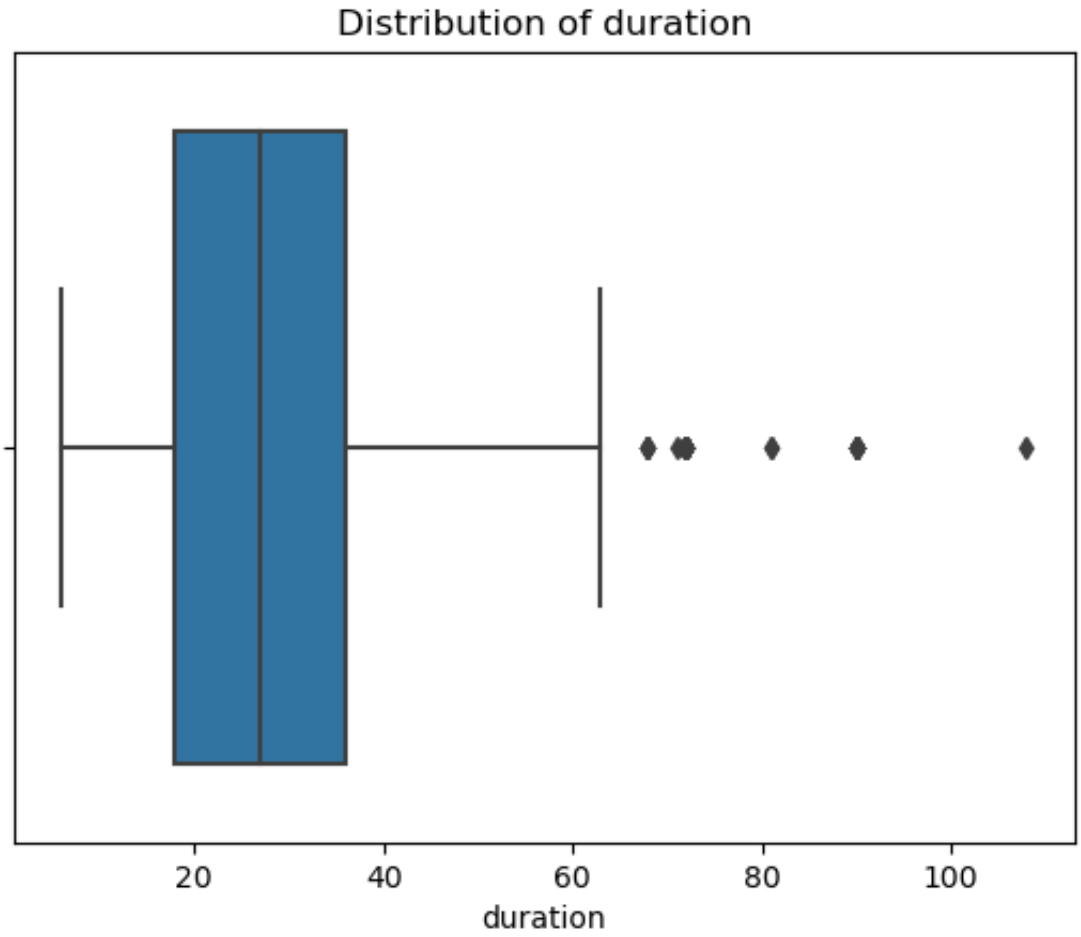
Checking for Outliers

In [10]:
```
loan_data.duration.describe()
```

Out[10]:
```
count    1000.000000
mean       31.449000
std        18.055564
min         6.000000
25%        18.000000
50%        27.000000
75%        36.000000
max       108.000000
Name: duration, dtype: float64
```

In [11]:
```
sns.boxplot(loan_data.duration)
plt.title('Distribution of duration')
plt.show
```

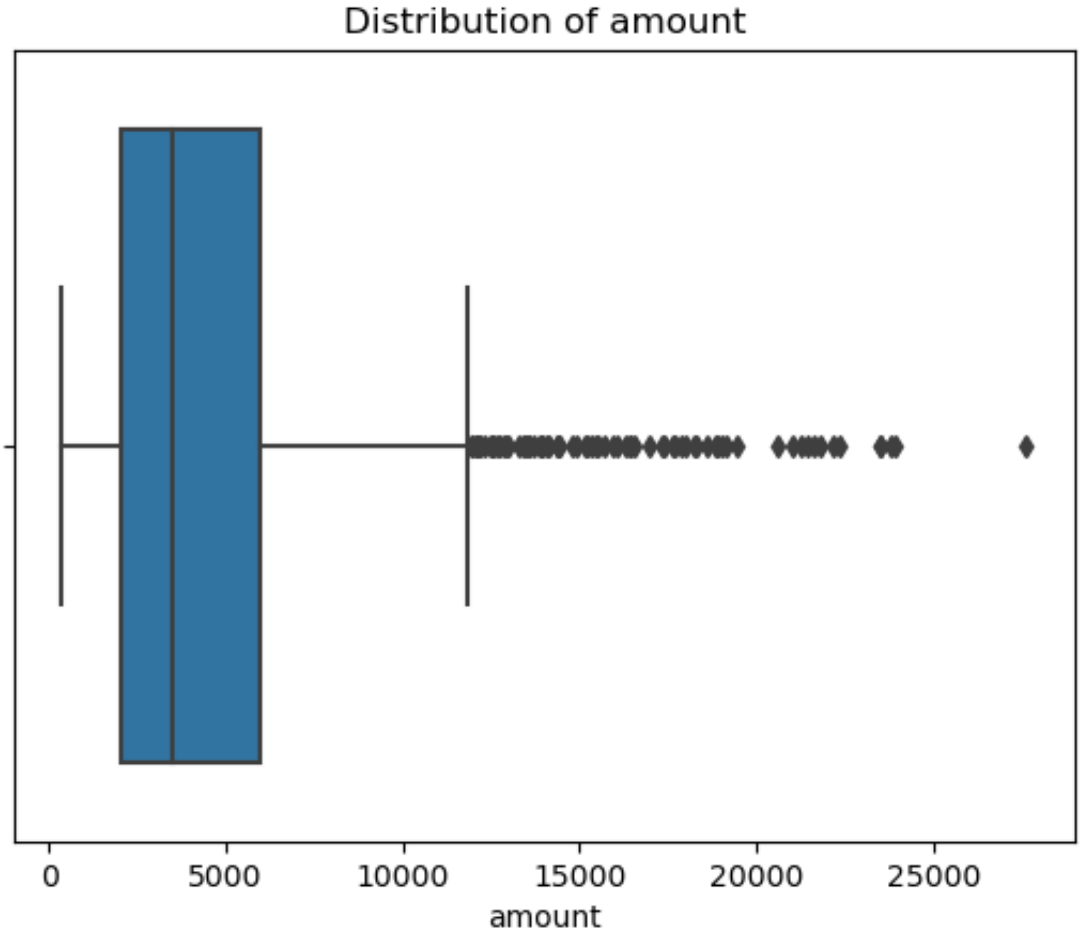
Out[11]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [12]:
```
# Here we get the outliers after the maximum value of 63. But here the du
# person so we don't remove it from the data.
```

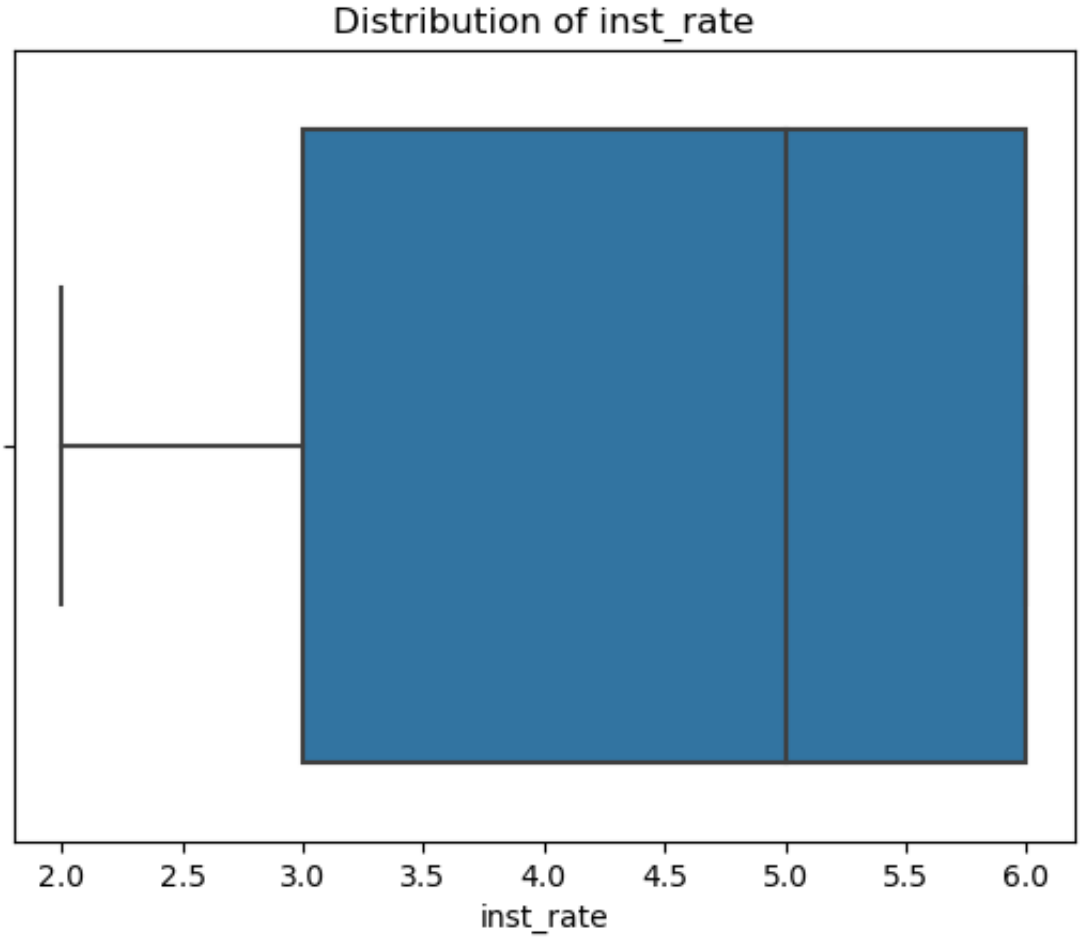
In [13]:
```
sns.boxplot(loan_data.amount)
plt.title('Distribution of amount')
plt.show
```

Out[13]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [14]:
```
sns.boxplot(loan_data.inst_rate)
plt.title('Distribution of inst_rate')
plt.show
```

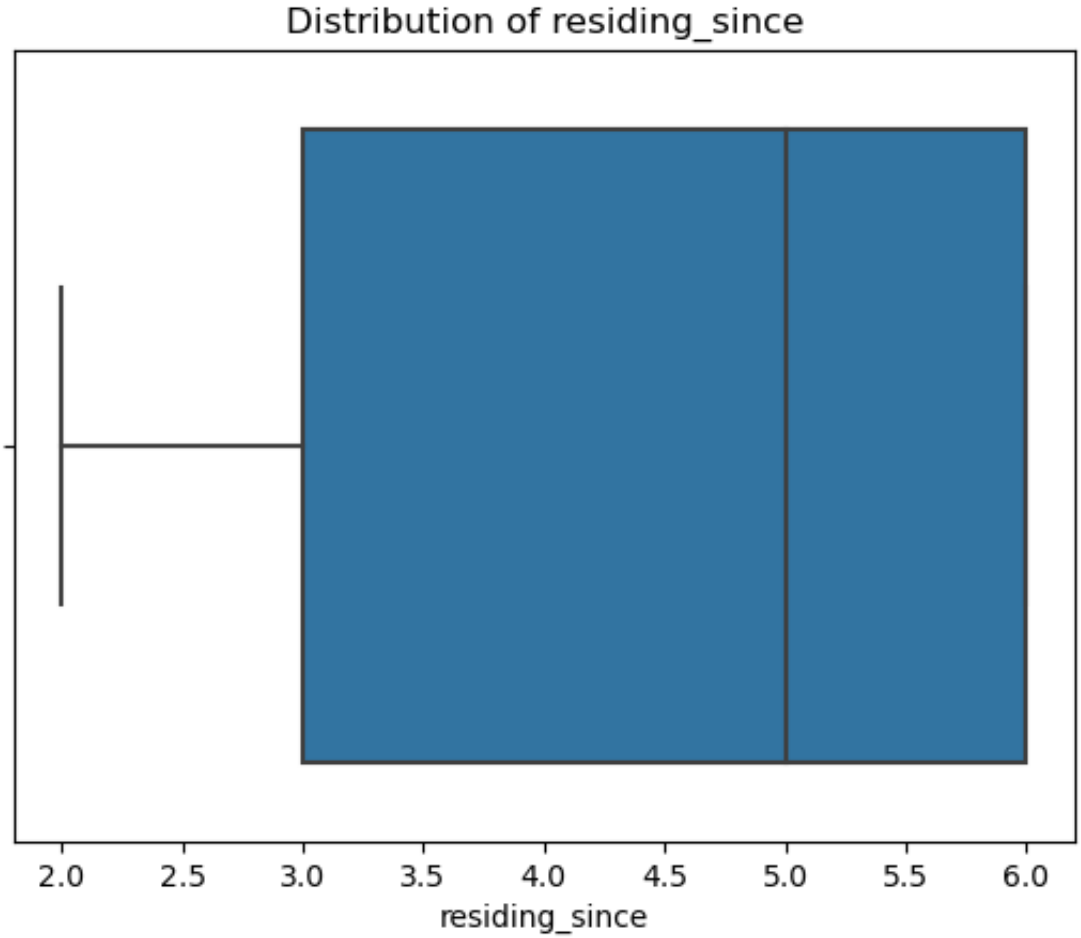
Out[14]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [15]:
```
# Here it is Right skewed and 75% vlaue is the maximum value
```

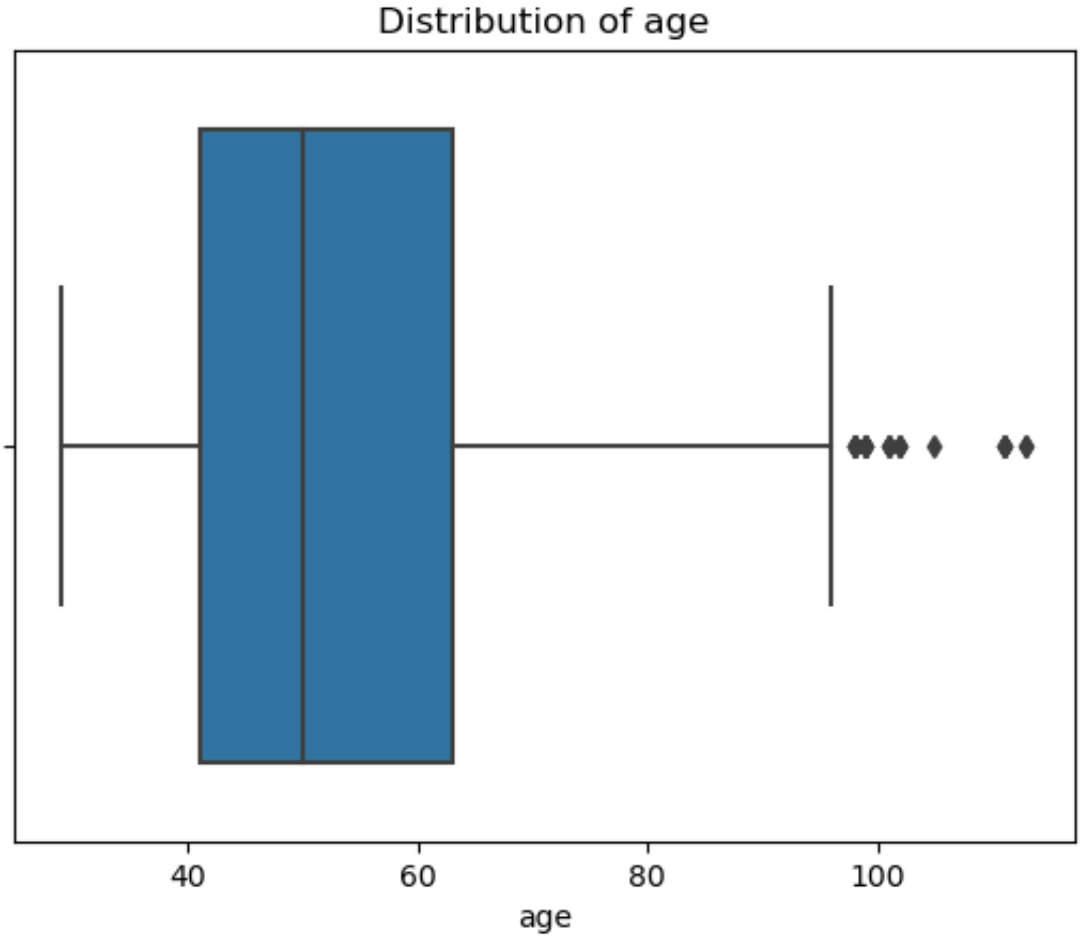
In [16]:
```
sns.boxplot(loan_data.residing_since)
plt.title('Distribution of residing_since')
plt.show
```

Out[16]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [17]:
```
sns.boxplot(loan_data.age)
plt.title('Distribution of age')
plt.show
```

Out[17]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [18]:
```
# Here we have some outliers so we have to remove them
```

In [19]:
```
loan_data = loan_data.drop(loan_data[loan_data['age']> 95]. index)
```

In [20]:
```
loan_data.head()
```

Out[20]:

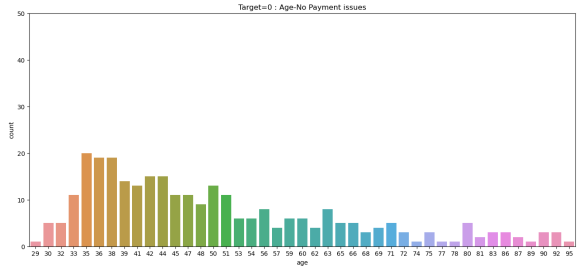
	checkin_acc	duration	credit_history	purpose	amount	svaing_acc	present_emp_sir
1	A12	72	A32	A43	8927	A61	A
2	A14	18	A34	A46	3144	A61	A
3	A11	63	A32	A42	11823	A61	A
4	A11	36	A33	A40	7305	A61	A
5	A14	54	A32	A46	13583	A65	A

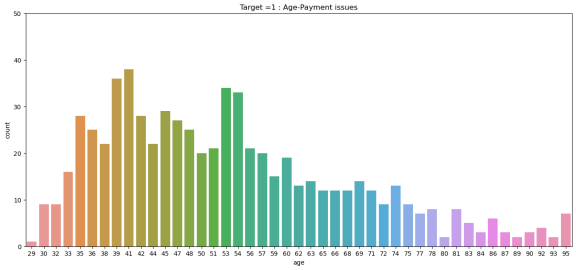
In [21]:
```
loan_data.shape
```

Out[21]:
```
(972, 21)
```

In [22]:
```
# Dividing the dataset into two dataset of  target=1(client with payment
target_1 = loan_data[loan_data['status']==1]
target_0 = loan_data[loan_data['status']==2]
```

In [23]:
```
# Numeric variable analysis for target_0 & target_1 dataframe

plt.figure(figsize = (35, 15))
plt.subplot(2, 2, 1)
plt.ylim(0,50)
plt.title('Target=0 : Age-No Payment issues')
sns.countplot(target_0['age'])

# subplot 2
plt.subplot(2, 2, 2)
plt.title('Target =1 : Age-Payment issues')
plt.ylim(0,50)
sns.countplot(target_1['age'])
plt.show()
```



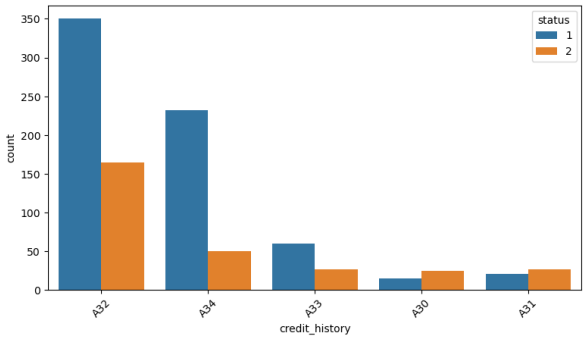


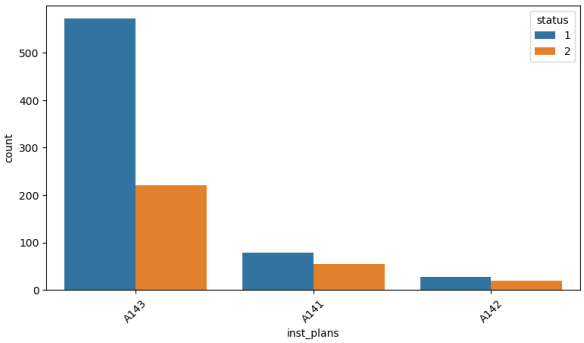
In [24]:
```
# Here we can see that the age group of 33-51 are able to make payemnt on
```

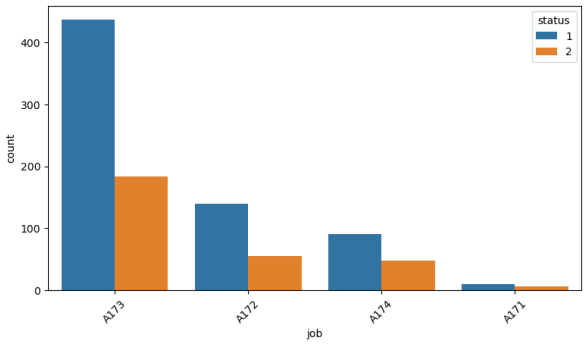
In [25]:
```
#Plot mutiple categorical columns with respect to Target column: Subplot

features = ['credit_history','inst_plans','job']
list(enumerate(features))
```

Out[25]:
```
[(0, 'credit_history'), (1, 'inst_plans'), (2, 'job')]
```

In [26]:
```
features = ['credit_history','inst_plans','job']
plt.figure(figsize = (20, 40))

plt.subplots_adjust(hspace=0.8)
for i in enumerate(features):
    plt.subplot(5, 2, i[0]+1)
    sns.countplot(x = i[1], hue = 'status', data = loan_data)
    plt.xticks(rotation = 45)
```

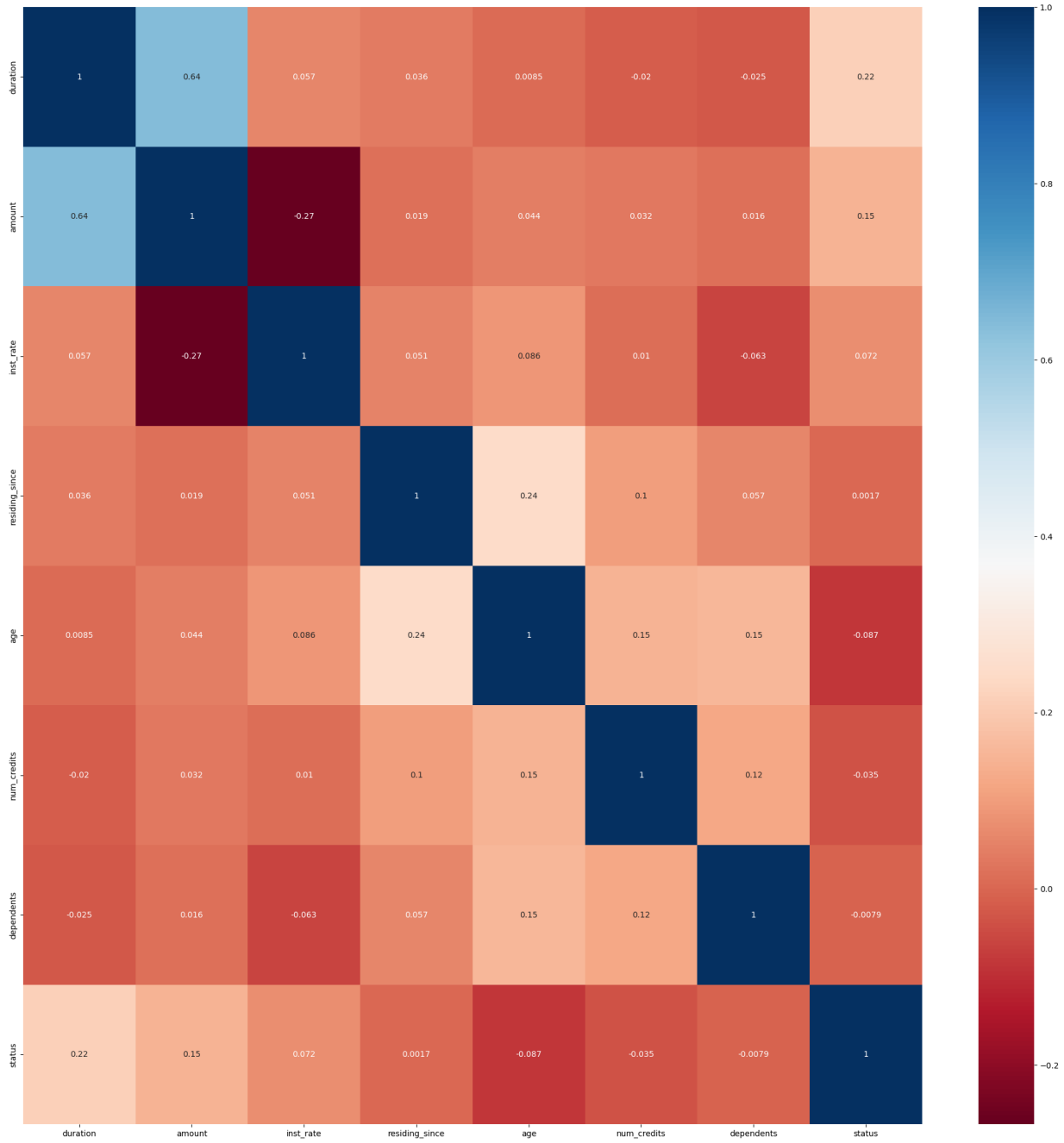






In [27]:
```
# Here we can see that whose credit history is of A32, A34 type are more
# Here in the instrest_plan graph we can see that the A143 type persons a
# Here in the job graph we can see that A173 are more likely to repay the
```

In [28]:
```
# Correlation coefficients to see which variable are highly correlated:

plt.figure(figsize = (25,25))
sns.heatmap(loan_data.corr(), annot =True, cmap="RdBu")
plt.show()
```


duration
amount
inst_rate
residing_since
age
num_credits
dependents
status
1 0.64 0.057 0.036 0.0085 -0.02 -0.025 0.22
0.64 1 -0.27 0.019 0.044 0.032 0.016 0.15
0.057 -0.27 1 0.051 0.086 0.01 -0.063 0.072
0.036 0.019 0.051 1 0.24 0.1 0.057 0.0017
0.0085 0.044 0.086 0.24 1 0.15 0.15 -0.087
-0.02 0.032 0.01 0.1 0.15 1 0.12 -0.035
-0.025 0.016 -0.063 0.057 0.15 0.12 1 -0.0079
0.22 0.15 0.072 0.0017 -0.087 -0.035 -0.0079 1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

In [29]:
```
# load statmodels functions
from statsmodels.stats.outliers_influence import variance_inflation_facto
from statsmodels.tools.tools import add_constant

# compute the vif for all given features
def compute_vif(considered_features):

    X = loan_data[considered_features]
    # the calculation of variance inflation requires a constant
    X['intercept'] = 1

    # create dataframe to store vif values
    vif = pd.DataFrame()
    vif["Variable"] = X.columns
    vif["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X
    vif = vif[vif['Variable']!='intercept']
    return vif
```

In [30]:
```
parameters = {'duration','amount','inst_rate','residing_since','age','num
```

In [31]:
```
# features to consider removing
considered_features = (parameters)


# compute vif
compute_vif(considered_features).sort_values('VIF', ascending=False)
```

Out[31]:

	Variable	VIF
1	amount	2.037778
6	duration	1.891109
0	inst_rate	1.213670
3	age	1.115159
5	residing_since	1.070869
2	dependents	1.041640
4	num_credits	1.041197

In [32]:
```
# Here VIF is less than 5 so we don't remove any variable
```

Using Label Encoder

In [33]:
```
for col in loan_data.select_dtypes(include=['object']).columns:
    print(f"{col}:{loan_data[col].unique()}")
```

```
checkin_acc:['A12' 'A14' 'A11' 'A13']
credit_history:['A32' 'A34' 'A33' 'A30' 'A31']
purpose:['A43' 'A46' 'A42' 'A40' 'A41' 'A49' 'A44' 'A45' 'A410' 'A48']
svaing_acc:['A61' 'A65' 'A63' 'A64' 'A62']
present_emp_since:['A73' 'A74' 'A75' 'A71' 'A72']
personal_status:['A92' 'A93' 'A91' 'A94']
other_debtors:['A101' 'A103' 'A102']
property:['A121' 'A122' 'A124' 'A123']
inst_plans:['A143' 'A141' 'A142']
housing:['A152' 'A153' 'A151']
job:['A173' 'A172' 'A174' 'A171']
telephone:['A191' 'A192']
foreign_worker:['A201' 'A202']
```

In [34]:

```
from sklearn import preprocessing
for col in loan_data.select_dtypes(include=['object']).columns:

    label_encoder = preprocessing.LabelEncoder()
    label_encoder.fit(loan_data[col].unique())
    loan_data[col]= label_encoder.transform (loan_data[col])
    print(f"{col}: {loan_data[col].unique()}")
```

```
checkin_acc: [1 3 0 2]
credit_history: [2 4 3 0 1]
purpose: [4 7 3 0 1 9 5 6 2 8]
svaing_acc: [0 4 2 3 1]
present_emp_since: [2 3 4 0 1]
personal_status: [1 2 0 3]
other_debtors: [0 2 1]
property: [0 1 3 2]
inst_plans: [2 0 1]
housing: [1 2 0]
job: [2 1 3 0]
telephone: [0 1]
foreign_worker: [0 1]
```

In [35]:

```
loan_data
```

Out[35]:

	checkin_acc	duration	credit_history	purpose	amount	svaing_acc	present_emp_
1	1	72	2	4	8927	0	
2	3	18	4	7	3144	0	
3	0	63	2	3	11823	0	
4	0	36	3	0	7305	0	
5	3	54	2	7	13583	4	
...	...	...	...	...	...	...	
995	3	18	2	3	2604	0	
996	0	45	2	1	5786	0	
997	3	18	2	4	1206	0	
998	0	68	2	4	2768	0	
999	1	68	4	1	6864	1	

972 rows × 21 columns

Train_Test Split

In [36]:
```
x=loan_data.drop(['status'], axis=1)
x
```

Out[36]:

	checkin_acc	duration	credit_history	purpose	amount	svaing_acc	present_emp_
1	1	72	2	4	8927	0	
2	3	18	4	7	3144	0	
3	0	63	2	3	11823	0	
4	0	36	3	0	7305	0	
5	3	54	2	7	13583	4	
...	...	...	...	...	...	...	
995	3	18	2	3	2604	0	
996	0	45	2	1	5786	0	
997	3	18	2	4	1206	0	
998	0	68	2	4	2768	0	
999	1	68	4	1	6864	1	

972 rows × 20 columns

In [37]:
```
y = loan_data['status']
y
```

Out[37]:
```
1      2
2      1
3      1
4      2
5      1
      ..
995    1
996    1
997    1
998    2
999    1
Name: status, Length: 972, dtype: int64
```

In [38]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y, test_size = 0.3, ra
```

In [39]:
```
x_train
```

Out[39]:

	checkin_acc	duration	credit_history	purpose	amount	svaing_acc	present_emp_
929	0	18	3	0	2016	0	
815	1	54	3	0	11148	0	
468	3	50	3	9	4146	0	
666	1	45	1	3	5244	3	
237	1	32	2	9	4151	1	
...	...	...	...	...	...	...	
108	0	36	2	3	11582	4	
279	3	36	2	4	2852	1	
885	0	18	2	7	1193	0	
447	1	11	2	4	3864	0	
104	3	18	2	1	3668	4	

680 rows × 20 columns

In [40]:
```
x_test
```

Out[40]:

	checkin_acc	duration	credit_history	purpose	amount	svaing_acc	present_emp_
789	0	60	4	7	8997	0	
71	3	11	4	4	1095	4	
879	3	45	4	4	10113	4	
497	3	36	4	3	2378	0	
661	0	18	2	0	1350	4	
...	...	...	...	...	...	...	
321	0	36	2	4	2907	0	
573	0	23	2	9	1209	0	
933	3	18	4	4	783	2	
750	0	9	2	3	642	0	
525	1	39	2	1	11949	0	

292 rows × 20 columns

Scaling the Data

In [41]:
```
# Here some values are so big that they will overpower the small data, so

from sklearn.preprocessing import StandardScaler
scaler= StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
```

In [42]:
```
x_train_scaled
```

Out[42]:
```
array([[-1.2646268 , -0.75901724,  0.40101944, ...,  2.28878398,
        -0.82150888, -0.19938744],
       [-0.46911688,  1.22801539,  0.40101944, ..., -0.43691323,
        -0.82150888, -0.19938744],
       [ 1.12190296,  1.00723398,  0.40101944, ..., -0.43691323,
         1.21727229, -0.19938744],
       ...,
       [-1.2646268 , -0.75901724, -0.51099802, ..., -0.43691323,
        -0.82150888, -0.19938744],
       [-0.46911688, -1.1453847 , -0.51099802, ..., -0.43691323,
        -0.82150888, -0.19938744],
       [ 1.12190296, -0.75901724, -0.51099802, ..., -0.43691323,
         1.21727229, -0.19938744]])
```

In [43]:
```
x_test_scaled
```

Out[43]:
```
array([[-1.2646268 ,  1.55918749,  1.31303691, ..., -0.43691323,
         1.21727229, -0.19938744],
       [ 1.12190296, -1.1453847 ,  1.31303691, ..., -0.43691323,
         1.21727229, -0.19938744],
       [ 1.12190296,  0.73125723,  1.31303691, ..., -0.43691323,
        -0.82150888, -0.19938744],
       ...,
       [ 1.12190296, -0.75901724,  1.31303691, ...,  2.28878398,
         1.21727229, -0.19938744],
       [-1.2646268 , -1.2557754 , -0.51099802, ..., -0.43691323,
         1.21727229, -0.19938744],
       [-0.46911688,  0.40008512, -0.51099802, ..., -0.43691323,
        -0.82150888, -0.19938744]])
```

Model Fitting - Logistic Regression

In [44]:
```
from sklearn.linear_model import LogisticRegression
log_reg = LogisticRegression(random_state = 0).fit(x_train_scaled,y_train
```

In [45]:
```
x_train_scale = log_reg.predict(x_train_scaled)
```

In [46]:
```
x_train_scale
```

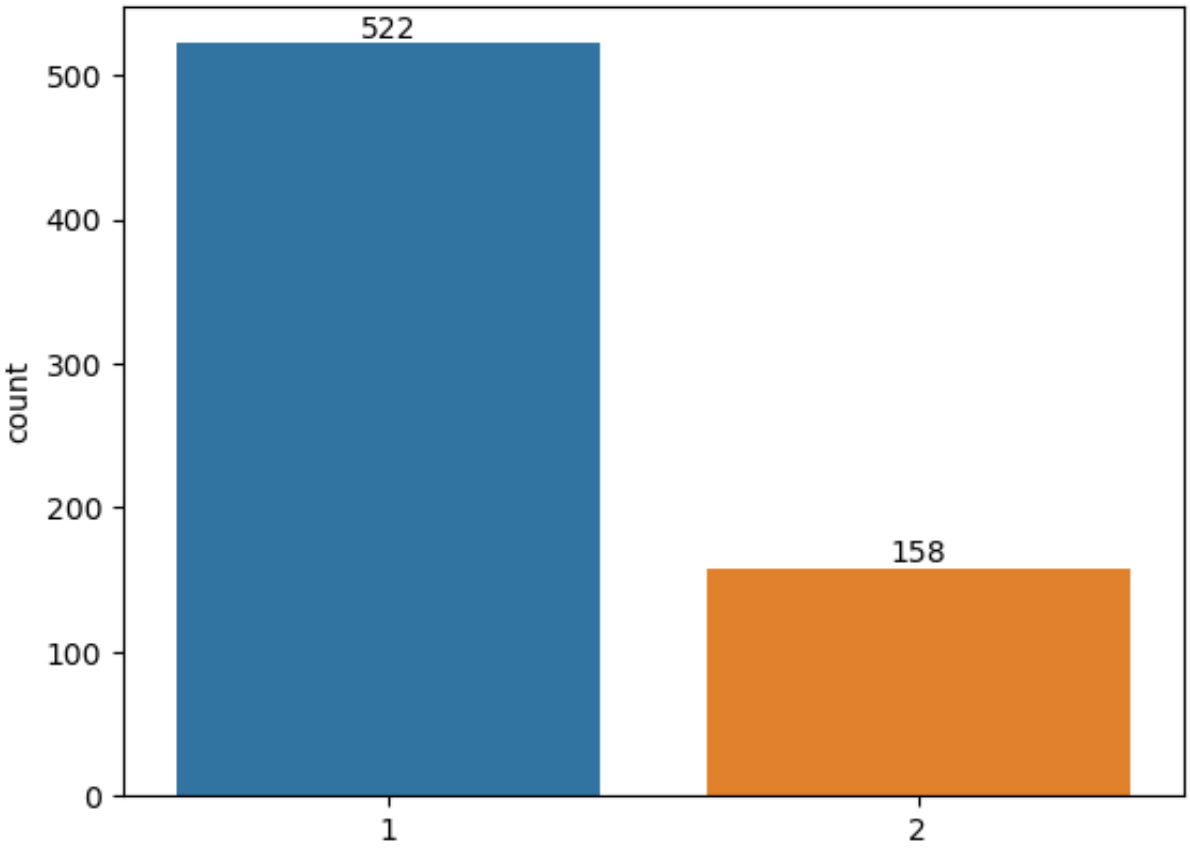
Out[46]:
```
array([2, 2, 1, 2, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 1,
       1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1,
       2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1,
       1, 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1,
       1, 1, 2, 2, 1, 2, 1, 2, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 2, 1,
       1, 1, 1, 1, 2, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1, 2, 1, 1, 2, 1, 1, 2,
       1, 1, 1, 1, 1, 2, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1,
       2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1,
       1, 2, 1, 2, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1,
       2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 1,
       1, 1, 1, 1, 2, 1, 1, 2, 2, 1, 1, 1, 2, 1, 1, 1, 1, 2, 2, 1, 1, 2,
       1, 2, 1, 1, 2, 1, 1, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1,
       1, 2, 2, 1, 1, 1, 1, 2, 1, 1, 2, 1, 2, 1, 1, 1, 1, 2, 2, 1, 1, 1,
       1, 1, 2, 2, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1])
```

In [47]:
```
from sklearn.metrics import accuracy_score, precision_score
y_pred = log_reg.predict(x_test_scaled)
print("Accuracy Score :", round(accuracy_score(y_test, y_pred)*100, 2), "
print("Precision Score :", round(precision_score(y_test, y_pred)*100, 2),
```

```
Accuracy Score : 75.0 %
Precision Score : 77.97 %
```

In [48]:
```
ax=sns.countplot(log_reg.predict(x_train_scaled))
ax.bar_label(ax.containers[0])
```

Out[48]:
```
[Text(0, 0, '522'), Text(0, 0, '158')]
```



Model Fitting - Decision Tree

In [49]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [50]:
```
#Post Prunning
treemodel = DecisionTreeClassifier()
```

In [51]:
```
treemodel.fit(x_train_scaled,y_train)
```

Out[51]:
```
DecisionTreeClassifier()
```

In [60]:
```
from sklearn import tree
plt.figure(figsize=(150,150))
tree.plot_tree(treemodel,filled = True)
```

Out[60]:
```
[Text(0.48672210300429186, 0.9705882352941176, 'X[0] <= -0.071\ngini = 0.
418\nsamples = 680\nvalue = [478, 202]'),
 Text(0.27360515021459225, 0.9117647058823529, 'X[1] <= -0.455\ngini = 0.
492\nsamples = 366\nvalue = [206, 160]'),
 Text(0.06598712446351931, 0.8529411764705882, 'X[2] <= -0.967\ngini = 0.
411\nsamples = 149\nvalue = [106, 43]'),
 Text(0.017167381974248927, 0.7941176470588235, 'X[16] <= -0.648\ngini =
0.337\nsamples = 14\nvalue = [3, 11]'),
 Text(0.008583690987124463, 0.7352941176470589, 'gini = 0.0\nsamples = 3\
nvalue = [3, 0]'),
 Text(0.02575107296137339, 0.7352941176470589, 'gini = 0.0\nsamples = 11\
nvalue = [0, 11]'),
 Text(0.1148068669527897, 0.7941176470588235, 'X[11] <= -0.827\ngini = 0.
362\nsamples = 135\nvalue = [103, 32]'),
 Text(0.04291845493562232, 0.7352941176470589, 'X[8] <= -1.662\ngini = 0.
18\nsamples = 60\nvalue = [54, 6]'),
 Text(0.034334763948497854, 0.6764705882352942, 'gini = 0.0\nsamples = 1\
nvalue = [0, 1]'),
 Text(0.05150214592274678, 0.6764705882352942, 'X[10] <= -1.191\ngini =
0.155\nsamples = 59\nvalue = [54, 5]'),
 Text(0.02575107296137339, 0.6176470588235294, 'X[3] <= -0.673\ngini = 0.
444\nsamples = 9\nvalue = [6, 3]'),
 Text(0.017167381974248927, 0.5588235294117647, 'gini = 0.0\nsamples = 2\
nvalue = [0, 2]'),
 Text(0.034334763948497854, 0.5588235294117647, 'X[18] <= 0.198\ngini =
0.245\nsamples = 7\nvalue = [6, 1]'),
 Text(0.02575107296137339, 0.5, 'gini = 0.0\nsamples = 6\nvalue = [6,
0]'),
 Text(0.04291845493562232, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [0,
1]'),
 Text(0.07725321888412018, 0.6176470588235294, 'X[3] <= -1.038\ngini = 0.
077\nsamples = 50\nvalue = [48, 2]'),
 Text(0.06866952789699571, 0.5588235294117647, 'X[9] <= 0.748\ngini = 0.2
08\nsamples = 17\nvalue = [15, 2]'),
 Text(0.060085836909871244, 0.5, 'X[12] <= -0.774\ngini = 0.117\nsamples
= 16\nvalue = [15, 1]'),
 Text(0.05150214592274678, 0.4411764705882353, 'X[0] <= -0.867\ngini = 0.
444\nsamples = 3\nvalue = [2, 1]'),
 Text(0.04291845493562232, 0.38235294117647056, 'gini = 0.0\nsamples = 1\
nvalue = [0, 1]'),
 Text(0.060085836909871244, 0.38235294117647056, 'gini = 0.0\nsamples = 2
\nvalue = [2, 0]'),
 Text(0.06866952789699571, 0.4411764705882353, 'gini = 0.0\nsamples = 13\
nvalue = [13, 0]'),
 Text(0.07725321888412018, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [0,
1]'),
 Text(0.08583690987124463, 0.5588235294117647, 'gini = 0.0\nsamples = 33\
nvalue = [33, 0]'),
 Text(0.18669527896995708, 0.7352941176470589, 'X[4] <= -0.54\ngini = 0.4
53\nsamples = 75\nvalue = [49, 26]'),
 Text(0.14163090128755365, 0.6764705882352942, 'X[1] <= -1.035\ngini = 0.
499\nsamples = 46\nvalue = [24, 22]'),
```

```
 Text(0.11158798283261803, 0.6176470588235294, 'X[15] <= 0.079\ngini = 0.
278\nsamples = 12\nvalue = [10, 2]'),
 Text(0.10300429184549356, 0.5588235294117647, 'X[7] <= 0.555\ngini = 0.4
8\nsamples = 5\nvalue = [3, 2]'),
 Text(0.0944206008583691, 0.5, 'gini = 0.0\nsamples = 2\nvalue = [2,
0]'),
 Text(0.11158798283261803, 0.5, 'X[6] <= 0.118\ngini = 0.444\nsamples = 3
\nvalue = [1, 2]'),
 Text(0.10300429184549356, 0.4411764705882353, 'gini = 0.0\nsamples = 2\n
value = [0, 2]'),
 Text(0.12017167381974249, 0.4411764705882353, 'gini = 0.0\nsamples = 1\n
value = [1, 0]'),
 Text(0.12017167381974249, 0.5588235294117647, 'gini = 0.0\nsamples = 7\n
value = [7, 0]'),
 Text(0.17167381974248927, 0.6176470588235294, 'X[4] <= -0.811\ngini = 0.
484\nsamples = 34\nvalue = [14, 20]'),
 Text(0.15450643776824036, 0.5588235294117647, 'X[12] <= -0.969\ngini =
0.165\nsamples = 11\nvalue = [1, 10]'),
 Text(0.1459227467811159, 0.5, 'X[6] <= 0.963\ngini = 0.5\nsamples = 2\nv
alue = [1, 1]'),
 Text(0.13733905579399142, 0.4411764705882353, 'gini = 0.0\nsamples = 1\n
value = [0, 1]'),
 Text(0.15450643776824036, 0.4411764705882353, 'gini = 0.0\nsamples = 1\n
value = [1, 0]'),
 Text(0.1630901287553648, 0.5, 'gini = 0.0\nsamples = 9\nvalue = [0,
9]'),
 Text(0.1888412017167382, 0.5588235294117647, 'X[6] <= -0.727\ngini = 0.4
91\nsamples = 23\nvalue = [13, 10]'),
 Text(0.18025751072961374, 0.5, 'gini = 0.0\nsamples = 5\nvalue = [0,
5]'),
 Text(0.19742489270386265, 0.5, 'X[3] <= 0.786\ngini = 0.401\nsamples = 1
8\nvalue = [13, 5]'),
 Text(0.18025751072961374, 0.4411764705882353, 'X[12] <= -1.359\ngini =
0.26\nsamples = 13\nvalue = [11, 2]'),
 Text(0.17167381974248927, 0.38235294117647056, 'gini = 0.0\nsamples = 1\
nvalue = [0, 1]'),
 Text(0.1888412017167382, 0.38235294117647056, 'X[3] <= -0.673\ngini = 0.
153\nsamples = 12\nvalue = [11, 1]'),
 Text(0.18025751072961374, 0.3235294117647059, 'X[6] <= 0.118\ngini = 0.4
44\nsamples = 3\nvalue = [2, 1]'),
 Text(0.17167381974248927, 0.2647058823529412, 'gini = 0.0\nsamples = 2\n
value = [2, 0]'),
 Text(0.1888412017167382, 0.2647058823529412, 'gini = 0.0\nsamples = 1\nv
alue = [0, 1]'),
 Text(0.19742489270386265, 0.3235294117647059, 'gini = 0.0\nsamples = 9\n
value = [9, 0]'),
 Text(0.2145922746781116, 0.4411764705882353, 'X[2] <= -0.055\ngini = 0.4
8\nsamples = 5\nvalue = [2, 3]'),
 Text(0.20600858369098712, 0.38235294117647056, 'gini = 0.0\nsamples = 2\
nvalue = [2, 0]'),
 Text(0.22317596566523606, 0.38235294117647056, 'gini = 0.0\nsamples = 3\
nvalue = [0, 3]'),
 Text(0.2317596566523605, 0.6764705882352942, 'X[6] <= 0.963\ngini = 0.23
8\nsamples = 29\nvalue = [25, 4]'),
 Text(0.2145922746781116, 0.6176470588235294, 'X[4] <= 0.19\ngini = 0.077
\nsamples = 25\nvalue = [24, 1]'),
 Text(0.20600858369098712, 0.5588235294117647, 'gini = 0.0\nsamples = 21\
```

```
nvalue = [21, 0]'),
 Text(0.22317596566523606, 0.5588235294117647, 'X[18] <= 0.198\ngini = 0.
375\nsamples = 4\nvalue = [3, 1]'),
 Text(0.2145922746781116, 0.5, 'gini = 0.0\nsamples = 3\nvalue = [3,
0]'),
 Text(0.2317596566523605, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [0,
1]'),
 Text(0.24892703862660945, 0.6176470588235294, 'X[15] <= 2.252\ngini = 0.
375\nsamples = 4\nvalue = [1, 3]'),
 Text(0.24034334763948498, 0.5588235294117647, 'gini = 0.0\nsamples = 3\n
value = [0, 3]'),
 Text(0.2575107296137339, 0.5588235294117647, 'gini = 0.0\nsamples = 1\nv
alue = [1, 0]'),
 Text(0.4812231759656652, 0.8529411764705882, 'X[5] <= 0.919\ngini = 0.49
7\nsamples = 217\nvalue = [100, 117]'),
 Text(0.434549356223176, 0.7941176470588235, 'X[1] <= 2.194\ngini = 0.476
\nsamples = 174\nvalue = [68, 106]'),
 Text(0.37553648068669526, 0.7352941176470589, 'X[18] <= 0.198\ngini = 0.
493\nsamples = 150\nvalue = [66, 84]'),
 Text(0.31759656652360513, 0.6764705882352942, 'X[4] <= 0.316\ngini = 0.4
5\nsamples = 82\nvalue = [28, 54]'),
 Text(0.2918454935622318, 0.6176470588235294, 'X[15] <= 0.079\ngini = 0.3
87\nsamples = 61\nvalue = [16, 45]'),
 Text(0.27467811158798283, 0.5588235294117647, 'X[8] <= 1.163\ngini = 0.4
44\nsamples = 45\nvalue = [15, 30]'),
 Text(0.26609442060085836, 0.5, 'X[19] <= 2.408\ngini = 0.422\nsamples =
43\nvalue = [13, 30]'),
 Text(0.2575107296137339, 0.4411764705882353, 'X[12] <= 1.76\ngini = 0.39
3\nsamples = 41\nvalue = [11, 30]'),
 Text(0.24892703862660945, 0.38235294117647056, 'X[0] <= -0.867\ngini =
0.375\nsamples = 40\nvalue = [10, 30]'),
 Text(0.2317596566523605, 0.3235294117647059, 'X[14] <= 1.088\ngini = 0.2
6\nsamples = 26\nvalue = [4, 22]'),
 Text(0.22317596566523606, 0.2647058823529412, 'X[2] <= 0.401\ngini = 0.2
11\nsamples = 25\nvalue = [3, 22]'),
 Text(0.2145922746781116, 0.20588235294117646, 'X[6] <= 0.118\ngini = 0.1
53\nsamples = 24\nvalue = [2, 22]'),
 Text(0.20600858369098712, 0.14705882352941177, 'gini = 0.0\nsamples = 18
\nvalue = [0, 18]'),
 Text(0.22317596566523606, 0.14705882352941177, 'X[7] <= -0.081\ngini =
0.444\nsamples = 6\nvalue = [2, 4]'),
 Text(0.2145922746781116, 0.08823529411764706, 'gini = 0.0\nsamples = 2\n
value = [2, 0]'),
 Text(0.2317596566523605, 0.08823529411764706, 'gini = 0.0\nsamples = 4\n
value = [0, 4]'),
 Text(0.2317596566523605, 0.20588235294117646, 'gini = 0.0\nsamples = 1\n
value = [1, 0]'),
 Text(0.24034334763948498, 0.2647058823529412, 'gini = 0.0\nsamples = 1\n
value = [1, 0]'),
 Text(0.26609442060085836, 0.3235294117647059, 'X[1] <= -0.124\ngini = 0.
49\nsamples = 14\nvalue = [6, 8]'),
 Text(0.2575107296137339, 0.2647058823529412, 'gini = 0.0\nsamples = 2\nv
alue = [2, 0]'),
 Text(0.27467811158798283, 0.2647058823529412, 'X[12] <= -0.32\ngini = 0.
444\nsamples = 12\nvalue = [4, 8]'),
 Text(0.26609442060085836, 0.20588235294117646, 'X[9] <= 1.804\ngini = 0.
5\nsamples = 8\nvalue = [4, 4]'),
```

```
 Text(0.2575107296137339, 0.14705882352941177, 'X[3] <= -0.308\ngini = 0.
444\nsamples = 6\nvalue = [4, 2]'),
 Text(0.24892703862660945, 0.08823529411764706, 'X[10] <= -1.191\ngini =
0.444\nsamples = 3\nvalue = [1, 2]'),
 Text(0.24034334763948498, 0.029411764705882353, 'gini = 0.0\nsamples = 1
\nvalue = [1, 0]'),
 Text(0.2575107296137339, 0.029411764705882353, 'gini = 0.0\nsamples = 2\
nvalue = [0, 2]'),
 Text(0.26609442060085836, 0.08823529411764706, 'gini = 0.0\nsamples = 3\
nvalue = [3, 0]'),
 Text(0.27467811158798283, 0.14705882352941177, 'gini = 0.0\nsamples = 2\
nvalue = [0, 2]'),
 Text(0.2832618025751073, 0.20588235294117646, 'gini = 0.0\nsamples = 4\n
value = [0, 4]'),
 Text(0.26609442060085836, 0.38235294117647056, 'gini = 0.0\nsamples = 1\
nvalue = [1, 0]'),
 Text(0.27467811158798283, 0.4411764705882353, 'gini = 0.0\nsamples = 2\n
value = [2, 0]'),
 Text(0.2832618025751073, 0.5, 'gini = 0.0\nsamples = 2\nvalue = [2,
0]'),
 Text(0.3090128755364807, 0.5588235294117647, 'X[7] <= -1.353\ngini = 0.1
17\nsamples = 16\nvalue = [1, 15]'),
 Text(0.30042918454935624, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [1,
0]'),
 Text(0.31759656652360513, 0.5, 'gini = 0.0\nsamples = 15\nvalue = [0, 1
5]'),
 Text(0.34334763948497854, 0.6176470588235294, 'X[4] <= 0.357\ngini = 0.4
9\nsamples = 21\nvalue = [12, 9]'),
 Text(0.33476394849785407, 0.5588235294117647, 'gini = 0.0\nsamples = 3\n
value = [3, 0]'),
 Text(0.351931330472103, 0.5588235294117647, 'X[4] <= 0.718\ngini = 0.5\n
samples = 18\nvalue = [9, 9]'),
 Text(0.33476394849785407, 0.5, 'X[4] <= 0.492\ngini = 0.375\nsamples = 8
\nvalue = [2, 6]'),
 Text(0.3261802575107296, 0.4411764705882353, 'X[7] <= -1.353\ngini = 0.4
44\nsamples = 3\nvalue = [2, 1]'),
 Text(0.31759656652360513, 0.38235294117647056, 'gini = 0.0\nsamples = 1\
nvalue = [0, 1]'),
 Text(0.33476394849785407, 0.38235294117647056, 'gini = 0.0\nsamples = 2\
nvalue = [2, 0]'),
 Text(0.34334763948497854, 0.4411764705882353, 'gini = 0.0\nsamples = 5\n
value = [0, 5]'),
 Text(0.36909871244635195, 0.5, 'X[4] <= 1.61\ngini = 0.42\nsamples = 10\
nvalue = [7, 3]'),
 Text(0.3605150214592275, 0.4411764705882353, 'gini = 0.0\nsamples = 6\nv
alue = [6, 0]'),
 Text(0.3776824034334764, 0.4411764705882353, 'X[4] <= 3.729\ngini = 0.37
5\nsamples = 4\nvalue = [1, 3]'),
 Text(0.36909871244635195, 0.38235294117647056, 'gini = 0.0\nsamples = 3\
nvalue = [0, 3]'),
 Text(0.38626609442060084, 0.38235294117647056, 'gini = 0.0\nsamples = 1\
nvalue = [1, 0]'),
 Text(0.4334763948497854, 0.6764705882352942, 'X[1] <= 0.124\ngini = 0.49
3\nsamples = 68\nvalue = [38, 30]'),
 Text(0.40772532188841204, 0.6176470588235294, 'X[4] <= -0.621\ngini = 0.
291\nsamples = 17\nvalue = [14, 3]'),
 Text(0.39914163090128757, 0.5588235294117647, 'gini = 0.0\nsamples = 2\n
```

```
value = [0, 2]'),
 Text(0.41630901287553645, 0.5588235294117647, 'X[1] <= -0.345\ngini = 0.
124\nsamples = 15\nvalue = [14, 1]'),
 Text(0.407725321888412O4, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [0,
1]'),
 Text(0.4248927038626609, 0.5, 'gini = 0.0\nsamples = 14\nvalue = [14,
0]'),
 Text(0.4592274678111588, 0.6176470588235294, 'X[9] <= 0.748\ngini = 0.49
8\nsamples = 51\nvalue = [24, 27]'),
 Text(0.45064377682403434, 0.5588235294117647, 'X[4] <= 3.141\ngini = 0.5
\nsamples = 47\nvalue = [24, 23]'),
 Text(0.44206008583690987, 0.5, 'X[5] <= -0.364\ngini = 0.496\nsamples =
44\nvalue = [24, 20]'),
 Text(0.4206008583690987, 0.4411764705882353, 'X[12] <= -0.482\ngini = 0.
498\nsamples = 36\nvalue = [17, 19]'),
 Text(0.4034334763948498, 0.38235294117647056, 'X[14] <= 1.088\ngini = 0.
355\nsamples = 13\nvalue = [3, 10]'),
 Text(0.3948497854077253, 0.3235294117647059, 'X[6] <= 0.963\ngini = 0.16
5\nsamples = 11\nvalue = [1, 10]'),
 Text(0.38626609442060084, 0.2647058823529412, 'gini = 0.0\nsamples = 10\
nvalue = [0, 10]'),
 Text(0.4034334763948498, 0.2647058823529412, 'gini = 0.0\nsamples = 1\nv
alue = [1, 0]'),
 Text(0.41201716738197425, 0.3235294117647059, 'gini = 0.0\nsamples = 2\n
value = [2, 0]'),
 Text(0.43776824034334766, 0.38235294117647056, 'X[4] <= -0.345\ngini =
0.476\nsamples = 23\nvalue = [14, 9]'),
 Text(0.4291845493562232, 0.3235294117647059, 'gini = 0.0\nsamples = 3\nv
alue = [0, 3]'),
 Text(0.44635193133047213, 0.3235294117647059, 'X[3] <= -0.491\ngini = 0.
42\nsamples = 20\nvalue = [14, 6]'),
 Text(0.43776824034334766, 0.2647058823529412, 'gini = 0.0\nsamples = 9\n
value = [9, 0]'),
 Text(0.45493562231759654, 0.2647058823529412, 'X[1] <= 0.372\ngini = 0.4
96\nsamples = 11\nvalue = [5, 6]'),
 Text(0.44635193133047213, 0.20588235294117646, 'gini = 0.0\nsamples = 3\
nvalue = [0, 3]'),
 Text(0.463519313304721, 0.20588235294117646, 'X[7] <= -0.399\ngini = 0.4
69\nsamples = 8\nvalue = [5, 3]'),
 Text(0.45493562231759654, 0.14705882352941177, 'gini = 0.0\nsamples = 2\
nvalue = [0, 2]'),
 Text(0.4721030042918455, 0.14705882352941177, 'X[17] <= 0.926\ngini = 0.
278\nsamples = 6\nvalue = [5, 1]'),
 Text(0.463519313304721, 0.08823529411764706, 'gini = 0.0\nsamples = 3\nv
alue = [3, 0]'),
 Text(0.48068669527896996, 0.08823529411764706, 'X[4] <= -0.016\ngini =
0.444\nsamples = 3\nvalue = [2, 1]'),
 Text(0.4721030042918455, 0.029411764705882353, 'gini = 0.0\nsamples = 2\
nvalue = [2, 0]'),
 Text(0.4892703862660944, 0.029411764705882353, 'gini = 0.0\nsamples = 1\
nvalue = [0, 1]'),
 Text(0.463519313304721, 0.4411764705882353, 'X[6] <= -0.727\ngini = 0.21
9\nsamples = 8\nvalue = [7, 1]'),
 Text(0.45493562231759654, 0.38235294117647056, 'gini = 0.0\nsamples = 1\
nvalue = [0, 1]'),
 Text(0.4721030042918455, 0.38235294117647056, 'gini = 0.0\nsamples = 7\n
value = [7, 0]'),
```

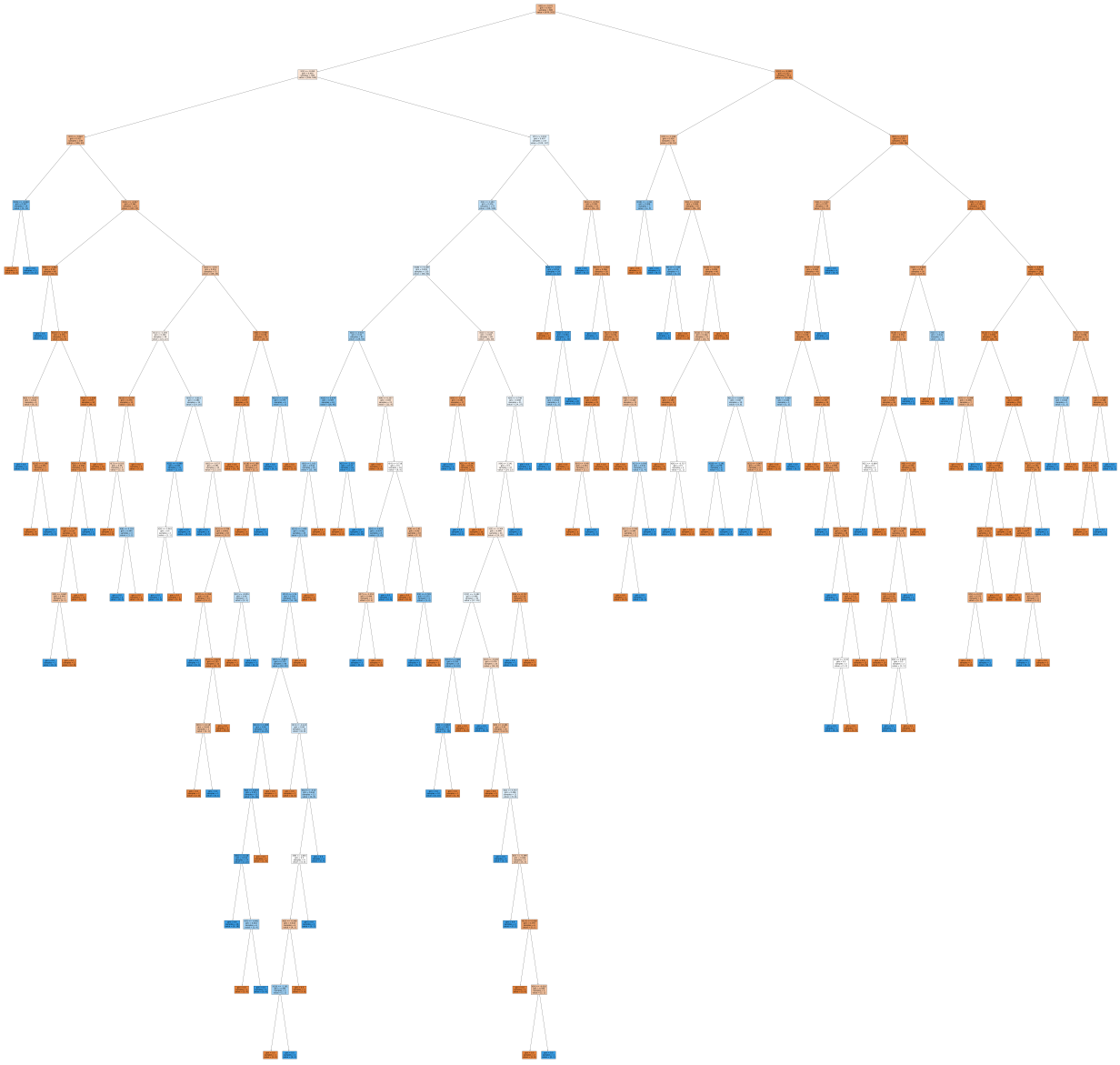
```
 Text(0.4592274678111588, 0.5, 'gini = 0.0\nsamples = 3\nvalue = [0,
3]'),
 Text(0.4678111587982833, 0.5588235294117647, 'gini = 0.0\nsamples = 4\nv
alue = [0, 4]'),
 Text(0.49356223175965663, 0.7352941176470589, 'X[6] <= -1.572\ngini = 0.
153\nsamples = 24\nvalue = [2, 22]'),
 Text(0.48497854077253216, 0.6764705882352942, 'gini = 0.0\nsamples = 1\n
value = [1, 0]'),
 Text(0.5021459227467812, 0.6764705882352942, 'X[4] <= 0.17\ngini = 0.083
\nsamples = 23\nvalue = [1, 22]'),
 Text(0.49356223175965663, 0.6176470588235294, 'X[4] <= 0.017\ngini = 0.4
44\nsamples = 3\nvalue = [1, 2]'),
 Text(0.48497854077253216, 0.5588235294117647, 'gini = 0.0\nsamples = 2\n
value = [0, 2]'),
 Text(0.5021459227467812, 0.5588235294117647, 'gini = 0.0\nsamples = 1\nv
alue = [1, 0]'),
 Text(0.5107296137339056, 0.6176470588235294, 'gini = 0.0\nsamples = 20\n
value = [0, 20]'),
 Text(0.5278969957081545, 0.7941176470588235, 'X[4] <= -0.559\ngini = 0.3
81\nsamples = 43\nvalue = [32, 11]'),
 Text(0.51931330472103, 0.7352941176470589, 'gini = 0.0\nsamples = 5\nval
ue = [0, 5]'),
 Text(0.5364806866952789, 0.7352941176470589, 'X[12] <= -1.262\ngini = 0.
266\nsamples = 38\nvalue = [32, 6]'),
 Text(0.5278969957081545, 0.6764705882352942, 'gini = 0.0\nsamples = 1\nv
alue = [0, 1]'),
 Text(0.5450643776824035, 0.6764705882352942, 'X[1] <= 0.98\ngini = 0.234
\nsamples = 37\nvalue = [32, 5]'),
 Text(0.5278969957081545, 0.6176470588235294, 'X[4] <= -0.471\ngini = 0.0
77\nsamples = 25\nvalue = [24, 1]'),
 Text(0.51931330472103, 0.5588235294117647, 'X[4] <= -0.48\ngini = 0.444\
nsamples = 3\nvalue = [2, 1]'),
 Text(0.5107296137339056, 0.5, 'gini = 0.0\nsamples = 2\nvalue = [2,
0]'),
 Text(0.5278969957081545, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [0,
1]'),
 Text(0.5364806866952789, 0.5588235294117647, 'gini = 0.0\nsamples = 22\n
value = [22, 0]'),
 Text(0.5622317596566524, 0.6176470588235294, 'X[6] <= 0.118\ngini = 0.44
4\nsamples = 12\nvalue = [8, 4]'),
 Text(0.5536480686695279, 0.5588235294117647, 'gini = 0.0\nsamples = 6\nv
alue = [6, 0]'),
 Text(0.5708154506437768, 0.5588235294117647, 'X[3] <= 0.056\ngini = 0.44
4\nsamples = 6\nvalue = [2, 4]'),
 Text(0.5622317596566524, 0.5, 'X[12] <= 0.395\ngini = 0.444\nsamples = 3
\nvalue = [2, 1]'),
 Text(0.5536480686695279, 0.4411764705882353, 'gini = 0.0\nsamples = 2\nv
alue = [2, 0]'),
 Text(0.5708154506437768, 0.4411764705882353, 'gini = 0.0\nsamples = 1\nv
alue = [0, 1]'),
 Text(0.5793991416309013, 0.5, 'gini = 0.0\nsamples = 3\nvalue = [0,
3]'),
 Text(0.6998390557939914, 0.9117647058823529, 'X[13] <= -0.288\ngini = 0.
232\nsamples = 314\nvalue = [272, 42]'),
 Text(0.5965665236051502, 0.8529411764705882, 'X[3] <= -1.038\ngini = 0.4
26\nsamples = 52\nvalue = [36, 16]'),
 Text(0.575107296137339, 0.7941176470588235, 'X[10] <= -1.191\ngini = 0.4
```

```
08\nsamples = 7\nvalue = [2, 5]'),
 Text(0.5665236051502146, 0.7352941176470589, 'gini = 0.0\nsamples = 2\nv
alue = [2, 0]'),
 Text(0.5836909871244635, 0.7352941176470589, 'gini = 0.0\nsamples = 5\nv
alue = [0, 5]'),
 Text(0.6180257510729614, 0.7941176470588235, 'X[1] <= -0.621\ngini = 0.3
69\nsamples = 45\nvalue = [34, 11]'),
 Text(0.6008583690987125, 0.7352941176470589, 'X[11] <= 1.06\ngini = 0.32
\nsamples = 5\nvalue = [1, 4]'),
 Text(0.592274678111588, 0.6764705882352942, 'gini = 0.0\nsamples = 4\nva
lue = [0, 4]'),
 Text(0.6094420600858369, 0.6764705882352942, 'gini = 0.0\nsamples = 1\nv
alue = [1, 0]'),
 Text(0.6351931330472103, 0.7352941176470589, 'X[10] <= -0.239\ngini = 0.
289\nsamples = 40\nvalue = [33, 7]'),
 Text(0.6266094420600858, 0.6764705882352942, 'X[12] <= -0.19\ngini = 0.4
44\nsamples = 21\nvalue = [14, 7]'),
 Text(0.5965665236051502, 0.6176470588235294, 'X[3] <= 1.151\ngini = 0.16
5\nsamples = 11\nvalue = [10, 1]'),
 Text(0.5879828326180258, 0.5588235294117647, 'gini = 0.0\nsamples = 9\nv
alue = [9, 0]'),
 Text(0.6051502145922747, 0.5588235294117647, 'X[6] <= -0.727\ngini = 0.5
\nsamples = 2\nvalue = [1, 1]'),
 Text(0.5965665236051502, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [0,
1]'),
 Text(0.6137339055793991, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [1,
0]'),
 Text(0.6566523605150214, 0.6176470588235294, 'X[12] <= 0.688\ngini = 0.4
8\nsamples = 10\nvalue = [4, 6]'),
 Text(0.6394849785407726, 0.5588235294117647, 'X[10] <= -1.191\ngini = 0.
278\nsamples = 6\nvalue = [1, 5]'),
 Text(0.630901287553648, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.648068669527897, 0.5, 'gini = 0.0\nsamples = 5\nvalue = [0, 5]'),
 Text(0.6738197424892703, 0.5588235294117647, 'X[2] <= -0.967\ngini = 0.3
75\nsamples = 4\nvalue = [3, 1]'),
 Text(0.6652360515021459, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [0,
1]'),
 Text(0.6824034334763949, 0.5, 'gini = 0.0\nsamples = 3\nvalue = [3,
0]'),
 Text(0.6437768240343348, 0.6764705882352942, 'gini = 0.0\nsamples = 19\n
value = [19, 0]'),
 Text(0.8031115879828327, 0.8529411764705882, 'X[6] <= -0.727\ngini = 0.1
79\nsamples = 262\nvalue = [236, 26]'),
 Text(0.7339055793991416, 0.7941176470588235, 'X[4] <= 1.214\ngini = 0.37
5\nsamples = 44\nvalue = [33, 11]'),
 Text(0.7253218884120172, 0.7352941176470589, 'X[9] <= 0.748\ngini = 0.28
9\nsamples = 40\nvalue = [33, 7]'),
 Text(0.7167381974248928, 0.6764705882352942, 'X[2] <= -0.967\ngini = 0.2
29\nsamples = 38\nvalue = [33, 5]'),
 Text(0.6995708154506438, 0.6176470588235294, 'X[8] <= -1.662\ngini = 0.4
44\nsamples = 3\nvalue = [1, 2]'),
 Text(0.6909871244635193, 0.5588235294117647, 'gini = 0.0\nsamples = 1\nv
alue = [1, 0]'),
 Text(0.7081545064377682, 0.5588235294117647, 'gini = 0.0\nsamples = 2\nv
alue = [0, 2]'),
 Text(0.7339055793991416, 0.6176470588235294, 'X[4] <= -0.538\ngini = 0.1
57\nsamples = 35\nvalue = [32, 3]'),
```

```
 Text(0.7253218884120172, 0.5588235294117647, 'gini = 0.0\nsamples = 16\n
value = [16, 0]'),
 Text(0.7424892703862661, 0.5588235294117647, 'X[1] <= -1.118\ngini = 0.2
66\nsamples = 19\nvalue = [16, 3]'),
 Text(0.7339055793991416, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [0,
1]'),
 Text(0.7510729613733905, 0.5, 'X[4] <= -0.447\ngini = 0.198\nsamples = 1
8\nvalue = [16, 2]'),
 Text(0.7424892703862661, 0.4411764705882353, 'gini = 0.0\nsamples = 1\nv
alue = [0, 1]'),
 Text(0.759656652360515, 0.4411764705882353, 'X[16] <= -0.648\ngini = 0.1
11\nsamples = 17\nvalue = [16, 1]'),
 Text(0.7510729613733905, 0.38235294117647056, 'X[12] <= -0.32\ngini = 0.
5\nsamples = 2\nvalue = [1, 1]'),
 Text(0.7424892703862661, 0.3235294117647059, 'gini = 0.0\nsamples = 1\nv
alue = [0, 1]'),
 Text(0.759656652360515, 0.3235294117647059, 'gini = 0.0\nsamples = 1\nva
lue = [1, 0]'),
 Text(0.7682403433476395, 0.38235294117647056, 'gini = 0.0\nsamples = 15\
nvalue = [15, 0]'),
 Text(0.7339055793991416, 0.6764705882352942, 'gini = 0.0\nsamples = 2\nv
alue = [0, 2]'),
 Text(0.7424892703862661, 0.7352941176470589, 'gini = 0.0\nsamples = 4\nv
alue = [0, 4]'),
 Text(0.8723175965665236, 0.7941176470588235, 'X[0] <= 0.724\ngini = 0.12
8\nsamples = 218\nvalue = [203, 15]'),
 Text(0.8197424892703863, 0.7352941176470589, 'X[17] <= 0.926\ngini = 0.3
2\nsamples = 30\nvalue = [24, 6]'),
 Text(0.8025751072961373, 0.6764705882352942, 'X[15] <= 2.252\ngini = 0.2
52\nsamples = 27\nvalue = [23, 4]'),
 Text(0.7939914163090128, 0.6176470588235294, 'X[4] <= -0.883\ngini = 0.2
04\nsamples = 26\nvalue = [23, 3]'),
 Text(0.776824034334764, 0.5588235294117647, 'X[1] <= -0.869\ngini = 0.5\
nsamples = 2\nvalue = [1, 1]'),
 Text(0.7682403433476395, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [0,
1]'),
 Text(0.7854077253218884, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [1,
0]'),
 Text(0.8111587982832618, 0.5588235294117647, 'X[6] <= 0.118\ngini = 0.15
3\nsamples = 24\nvalue = [22, 2]'),
 Text(0.8025751072961373, 0.5, 'X[14] <= 1.088\ngini = 0.26\nsamples = 13
\nvalue = [11, 2]'),
 Text(0.7939914163090128, 0.4411764705882353, 'X[1] <= 0.731\ngini = 0.15
3\nsamples = 12\nvalue = [11, 1]'),
 Text(0.7854077253218884, 0.38235294117647056, 'gini = 0.0\nsamples = 10\
nvalue = [10, 0]'),
 Text(0.8025751072961373, 0.38235294117647056, 'X[4] <= 0.622\ngini = 0.5
\nsamples = 2\nvalue = [1, 1]'),
 Text(0.7939914163090128, 0.3235294117647059, 'gini = 0.0\nsamples = 1\nv
alue = [0, 1]'),
 Text(0.8111587982832618, 0.3235294117647059, 'gini = 0.0\nsamples = 1\nv
alue = [1, 0]'),
 Text(0.8111587982832618, 0.4411764705882353, 'gini = 0.0\nsamples = 1\nv
alue = [0, 1]'),
 Text(0.8197424892703863, 0.5, 'gini = 0.0\nsamples = 11\nvalue = [11,
0]'),
 Text(0.8111587982832618, 0.6176470588235294, 'gini = 0.0\nsamples = 1\nv
```

```
alue = [0, 1]'),
 Text(0.8369098712446352, 0.6764705882352942, 'X[1] <= -1.09\ngini = 0.44
4\nsamples = 3\nvalue = [1, 2]'),
 Text(0.8283261802575107, 0.6176470588235294, 'gini = 0.0\nsamples = 1\nv
alue = [1, 0]'),
 Text(0.8454935622317596, 0.6176470588235294, 'gini = 0.0\nsamples = 2\nv
alue = [0, 2]'),
 Text(0.924892703862661, 0.7352941176470589, 'X[16] <= 0.954\ngini = 0.09
1\nsamples = 188\nvalue = [179, 9]'),
 Text(0.8841201716738197, 0.6764705882352942, 'X[12] <= -1.164\ngini = 0.
058\nsamples = 166\nvalue = [161, 5]'),
 Text(0.8626609442060086, 0.6176470588235294, 'X[7] <= -0.399\ngini = 0.4
08\nsamples = 7\nvalue = [5, 2]'),
 Text(0.8540772532188842, 0.5588235294117647, 'gini = 0.0\nsamples = 5\nv
alue = [5, 0]'),
 Text(0.871244635193133, 0.5588235294117647, 'gini = 0.0\nsamples = 2\nva
lue = [0, 2]'),
 Text(0.9055793991416309, 0.6176470588235294, 'X[17] <= 0.926\ngini = 0.0
37\nsamples = 159\nvalue = [156, 3]'),
 Text(0.8884120171673819, 0.5588235294117647, 'X[16] <= -0.648\ngini = 0.
015\nsamples = 133\nvalue = [132, 1]'),
 Text(0.8798283261802575, 0.5, 'X[4] <= -0.732\ngini = 0.111\nsamples = 1
7\nvalue = [16, 1]'),
 Text(0.871244635193133, 0.4411764705882353, 'X[5] <= 0.277\ngini = 0.375
\nsamples = 4\nvalue = [3, 1]'),
 Text(0.8626609442060086, 0.38235294117647056, 'gini = 0.0\nsamples = 3\n
value = [3, 0]'),
 Text(0.8798283261802575, 0.38235294117647056, 'gini = 0.0\nsamples = 1\n
value = [0, 1]'),
 Text(0.8884120171673819, 0.4411764705882353, 'gini = 0.0\nsamples = 13\n
value = [13, 0]'),
 Text(0.8969957081545065, 0.5, 'gini = 0.0\nsamples = 116\nvalue = [116,
0]'),
 Text(0.9227467811158798, 0.5588235294117647, 'X[12] <= 1.825\ngini = 0.1
42\nsamples = 26\nvalue = [24, 2]'),
 Text(0.9141630901287554, 0.5, 'X[12] <= 1.143\ngini = 0.077\nsamples = 2
5\nvalue = [24, 1]'),
 Text(0.9055793991416309, 0.4411764705882353, 'gini = 0.0\nsamples = 21\n
value = [21, 0]'),
 Text(0.9227467811158798, 0.4411764705882353, 'X[3] <= -0.491\ngini = 0.3
75\nsamples = 4\nvalue = [3, 1]'),
 Text(0.9141630901287554, 0.38235294117647056, 'gini = 0.0\nsamples = 1\n
value = [0, 1]'),
 Text(0.9313304721030042, 0.38235294117647056, 'gini = 0.0\nsamples = 3\n
value = [3, 0]'),
 Text(0.9313304721030042, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [0,
1]'),
 Text(0.9656652360515021, 0.6764705882352942, 'X[12] <= -0.58\ngini = 0.2
98\nsamples = 22\nvalue = [18, 4]'),
 Text(0.9484978540772532, 0.6176470588235294, 'X[6] <= 0.118\ngini = 0.44
4\nsamples = 3\nvalue = [1, 2]'),
 Text(0.9399141630901288, 0.5588235294117647, 'gini = 0.0\nsamples = 2\nv
alue = [0, 2]'),
 Text(0.9570815450643777, 0.5588235294117647, 'gini = 0.0\nsamples = 1\nv
alue = [1, 0]'),
 Text(0.9828326180257511, 0.6176470588235294, 'X[4] <= 3.034\ngini = 0.18
8\nsamples = 19\nvalue = [17, 2]'),
```

```
 Text(0.9742489270386266, 0.5588235294117647, 'X[15] <= 2.252\ngini = 0.1
05\nsamples = 18\nvalue = [17, 1]'),
 Text(0.9656652360515021, 0.5, 'gini = 0.0\nsamples = 17\nvalue = [17,
0]'),
 Text(0.9828326180257511, 0.5, 'gini = 0.0\nsamples = 1\nvalue = [0,
1]'),
 Text(0.9914163090128756, 0.5588235294117647, 'gini = 0.0\nsamples = 1\nv
alue = [0, 1]')]
```



In [62]:
```
# Here the tree is very large so we can't fix the max depth function
# So we do pre-prunning
```

In [63]:
```
parameters  = {'criterion':['gini','entropy','log_loss'],
               'splitter':['best','random'],
               'max_depth':[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17],
               'max_features':['auto','sqrt','log2']}
```

In [64]:
```
from sklearn.model_selection import GridSearchCV
```

In [66]:
```
treemodel = DecisionTreeClassifier(max_depth=2)
cv = GridSearchCV(treemodel,param_grid=parameters,cv=5,scoring='accuracy'
```

In [67]:
```
cv.fit(x_train_scaled,y_train)
```

Out[67]:
```
GridSearchCV(cv=5, estimator=DecisionTreeClassifier(max_depth=2),
             param_grid={'criterion': ['gini', 'entropy', 'log_loss'],
                         'max_depth': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11,
12,
                                       13, 14, 15, 16, 17],
                         'max_features': ['auto', 'sqrt', 'log2'],
                         'splitter': ['best', 'random']},
             scoring='accuracy')
```

In [70]:
```
# Here it gave us the best paramert
cv.best_params_
```

Out[70]:
```
{'criterion': 'entropy',
 'max_depth': 6,
 'max_features': 'auto',
 'splitter': 'random'}
```

In [73]:
```
y_pred = cv.predict(x_test_scaled)
```

In [74]:
```
from sklearn.metrics import accuracy_score, classification_report
```

In [75]:
```
score = accuracy_score(y_pred,y_test)
```

In [76]:
```
score
```

Out[76]:
```
0.678082191780822
```

In [79]:
```
print(classification_report(y_pred,y_test))
```

```
              precision    recall  f1-score   support

           1       0.89      0.71      0.79       250
           2       0.22      0.48      0.30        42

    accuracy                           0.68       292
   macro avg       0.55      0.59      0.54       292
weighted avg       0.79      0.68      0.72       292
```

In [80]:
```
# Here the descision tree won't wrok well because we have a limited data
```

In [81]:
```
y_pred = cv.predict(x_train_scaled)
```

In [86]:
```
score = accuracy_score(y_pred,y_train)
```

In [87]:
```
score
```

Out[87]:
```
0.7735294117647059
```

In [ ]:
```
# Here it works well on training data but not on test data means overfitt
```