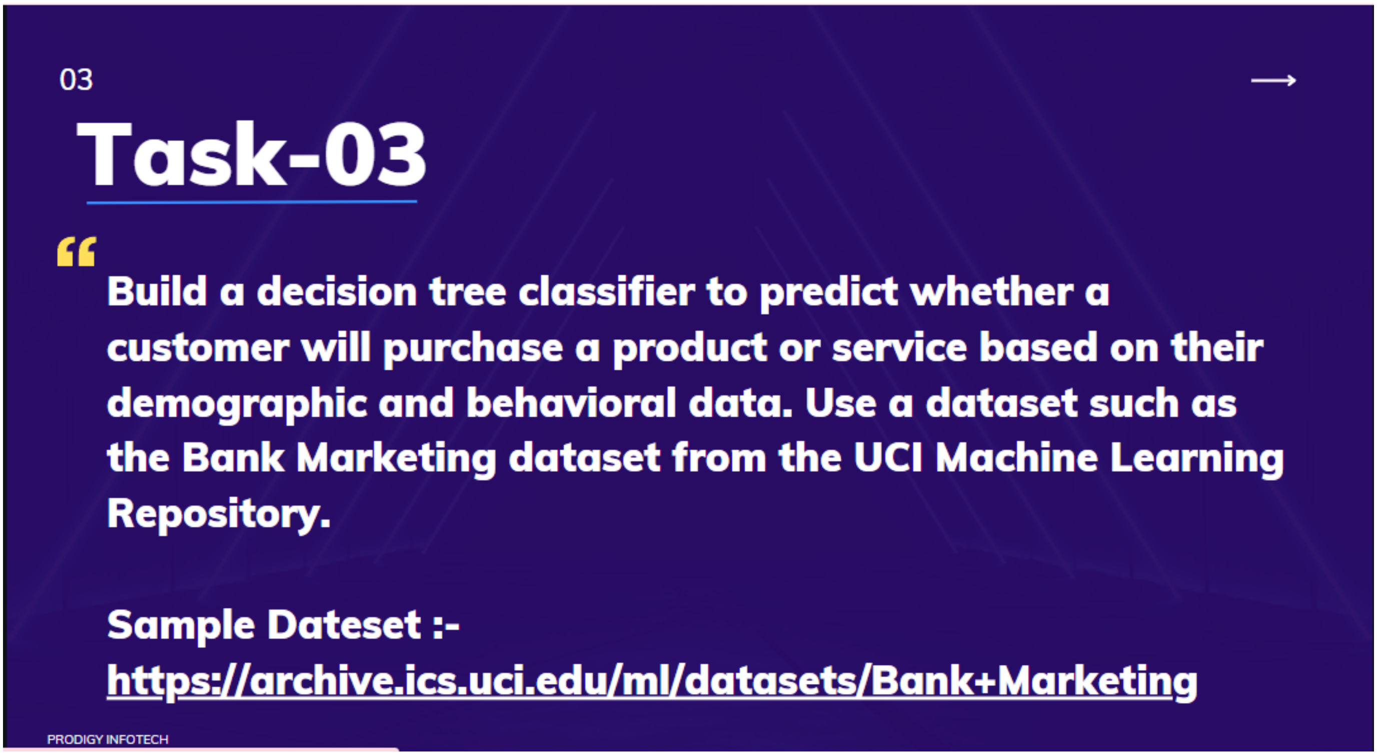


03

Task-03

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.

Sample Dateset :-
https://archive.ics.uci.edu/ml/datasets/Bank+Marketing

PRODIGY INFOTECH

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.model_selection  import train_test_split
from scipy.stats import chi2_contingency
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
import warnings
warnings.filterwarnings("ignore")
```

```
da =pd.read_csv("/content/bank-full.csv",sep=";")
```

Input variables:

bank client data:

1 - age (numeric)

2 - job : type of job (categorical: "admin.","unknown","unemployed","management","housemaid","entrepreneur","student", "blue-collar","self-employed","retired","technician","services")

3 - marital : marital status (categorical: "married","divorced","single"; note: "divorced" means divorced or widowed)

4 - education (categorical: "unknown","secondary","primary","tertiary")

5 - default: has credit in default? (binary: "yes","no")

6 - balance: average yearly balance, in euros (numeric)

7 - housing: has housing loan? (binary: "yes","no")

8 - loan: has personal loan? (binary: "yes","no")

related with the last contact of the current campaign:

9 - contact: contact communication type (categorical: "unknown","telephone","cellular")

10 - day: last contact day of the month (numeric)

11 - month: last contact month of year (categorical: "jan", "feb", "mar", ..., "nov", "dec")

12 - duration: last contact duration, in seconds (numeric)

other attributes:

13 - campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)

14 - pdays: number of days that passed by after the client was last contacted from a previous campaign (numeric, -1 means client was not previously contacted)

15 - previous: number of contacts performed before this campaign and for this client (numeric)

16 - poutcome: outcome of the previous marketing campaign (categorical: "unknown","other","failure","success")

Output variable (desired target):

17 - y - has the client subscribed a term deposit? (binary: "yes","no")

```
da.columns
```

```
Index(['age', 'job', 'marital', 'education', 'default', 'balance', 'housing',
       'loan', 'contact', 'day', 'month', 'duration', 'campaign', 'pdays',
       'previous', 'poutcome', 'y'],
      dtype='object')
```

```
da.head()
```

	age	job	marital	education	default	balance	housing	loan	contact	day
0	58	management	married	tertiary	no	2143	yes	no	unknown	5
1	44	technician	single	secondary	no	29	yes	no	unknown	5
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5
4	33	unknown	single	unknown	no	1	no	no	unknown	5

```
da.tail()
```

	age	job	marital	education	default	balance	housing	loan	contact
45206	51	technician	married	tertiary	no	825	no	no	cellular
45207	71	retired	divorced	primary	no	1729	no	no	cellular
45208	72	retired	married	secondary	no	5715	no	no	cellular
45209	57	blue-collar	married	secondary	no	668	no	no	telephone
45210	37	entrepreneur	married	secondary	no	2971	no	no	cellular

```
da.shape
```

```
(45211, 17)
```

```
da.isnull().sum()
```

```
age          0
job          0
marital      0
education    0
default      0
balance      0
housing      0
loan         0
contact      0
day          0
month        0
duration     0
campaign     0
pdays        0
previous     0
poutcome     0
y            0
dtype: int64
```

```
da.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 17 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   age        45211 non-null  int64 
 1   job        45211 non-null  object
 2   marital    45211 non-null  object
 3   education  45211 non-null  object
 4   default    45211 non-null  object
 5   balance    45211 non-null  int64 
 6   housing    45211 non-null  object
 7   loan       45211 non-null  object
 8   contact    45211 non-null  object
 9   day        45211 non-null  int64 
 10  month      45211 non-null  object
 11  duration   45211 non-null  int64 
 12  campaign   45211 non-null  int64 
 13  pdays      45211 non-null  int64 
 14  previous   45211 non-null  int64 
 15  poutcome   45211 non-null  object
 16  y          45211 non-null  object
dtypes: int64(7), object(10)
memory usage: 5.9+ MB
```

```
da.describe()
```

	age	balance	day	duration	campaign	pda
count	45211.000000	45211.000000	45211.000000	45211.000000	45211.000000	45211.0000
mean	40.936210	1362.272058	15.806419	258.163080	2.763841	40.1978
std	10.618762	3044.765829	8.322476	257.527812	3.098021	100.1287
min	18.000000	-8019.000000	1.000000	0.000000	1.000000	-1.0000
25%	33.000000	72.000000	8.000000	103.000000	1.000000	-1.0000
50%	39.000000	448.000000	16.000000	180.000000	2.000000	-1.0000
75%	48.000000	1428.000000	21.000000	319.000000	3.000000	-1.0000
max	95.000000	102127.000000	31.000000	4918.000000	63.000000	871.0000

```
da["y"].value_counts()
```

```
no     39922
yes     5289
Name: y, dtype: int64
```

```
yes=da.loc[da["y"]=="yes","y"].count()/len(da)*100
no = da.loc[da["y"]=="no","y"].count()/len(da)*100
print("Percentage of yes:", yes)
print("Percentage of no:", no)
```

```
Percentage of yes: 11.698480458295547
Percentage of no: 88.30151954170445
```

```
df=da.copy()
```

```
df.columns
```

```
Index(['age', 'job', 'marital', 'education', 'default', 'balance', 'housing',
       'loan', 'contact', 'day', 'month', 'duration', 'campaign', 'pdays',
       'previous', 'poutcome', 'y'],
      dtype='object')
```

```
df.shape
```

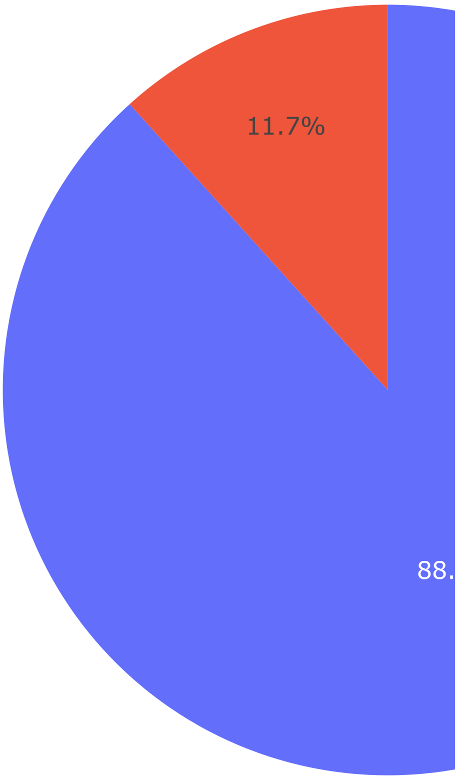
```
(45211, 17)
```

```
df["target"]=df["y"].map({"yes":1,"no":0})
```

```
df["target"].value_counts()
```

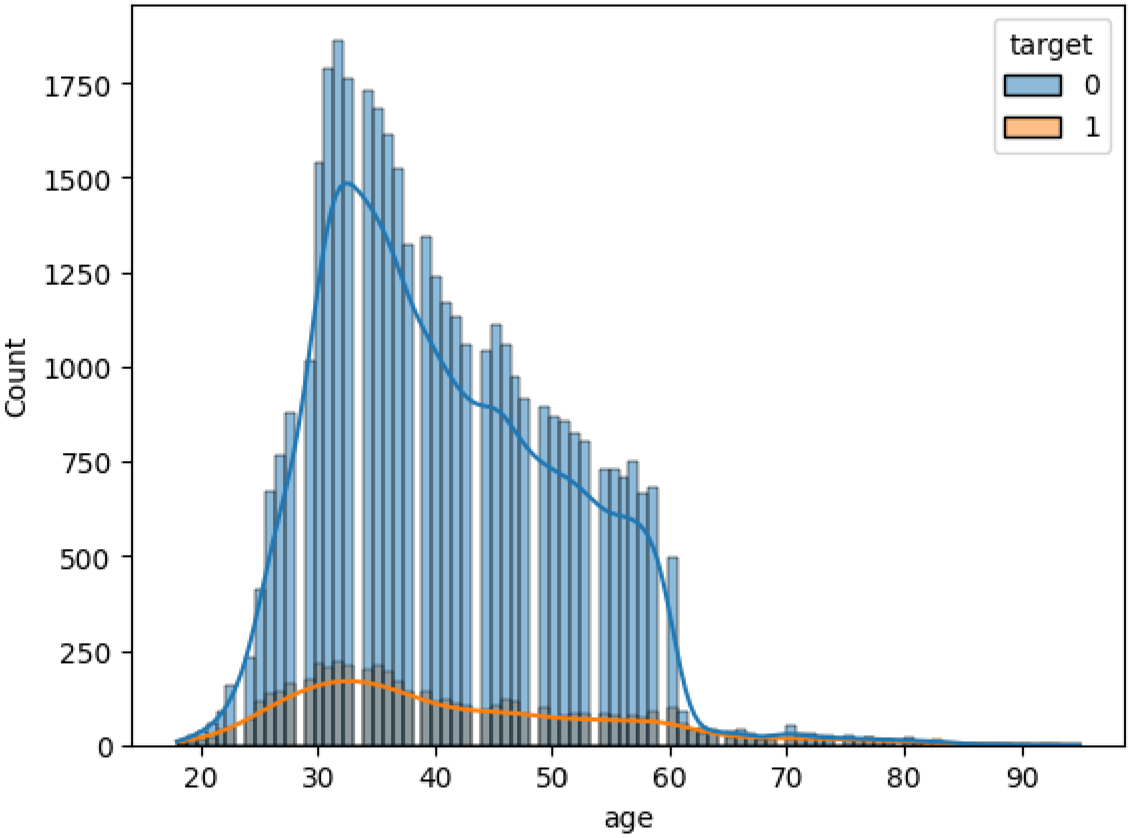
```
0    39922
1     5289
Name: target, dtype: int64
```

```
px.pie(df,"target")
```

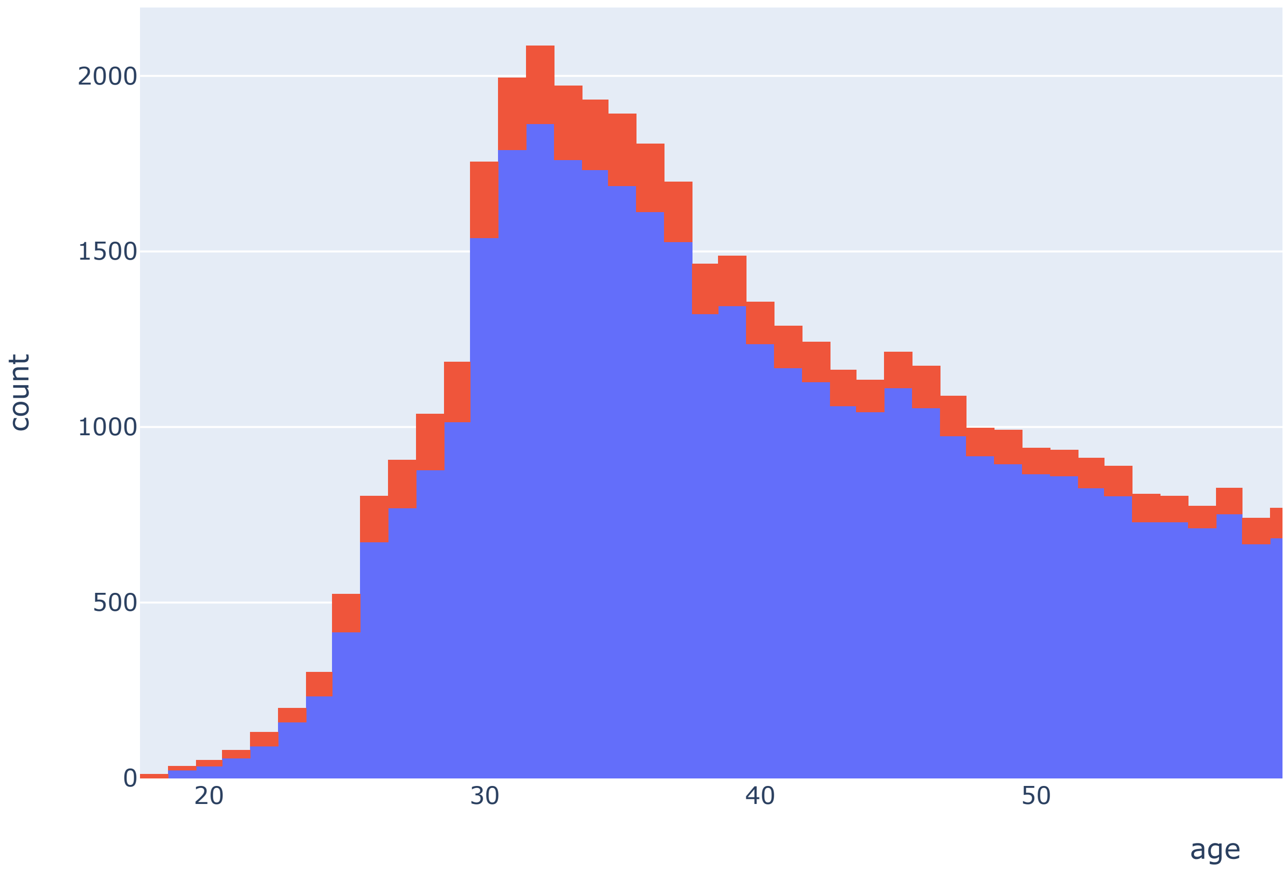


```
sns.histplot(data=df,x="age",kde=True,hue="target")
```

```
<Axes: xlabel='age', ylabel='Count'>
```

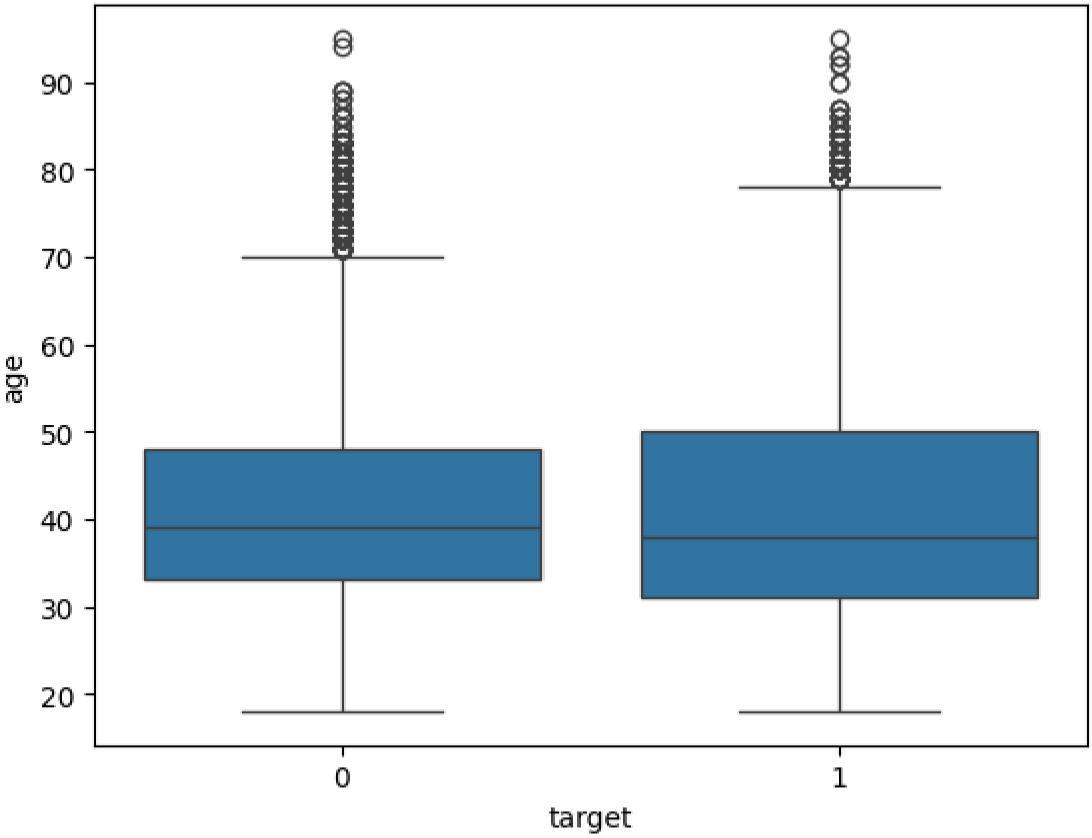


```
px.histogram(df,x="age",color="target")
```



```
sns.boxplot(data=df,y="age",x="target")
```

```
<Axes: xlabel='target', ylabel='age'>
```



```
df["job"].value_counts()
```

```
blue-collar      9732
management       9458
technician       7597
admin.           5171
services         4154
retired          2264
self-employed    1579
entrepreneur     1487
unemployed       1303
housemaid        1240
student           938
unknown           288
Name: job, dtype: int64
```

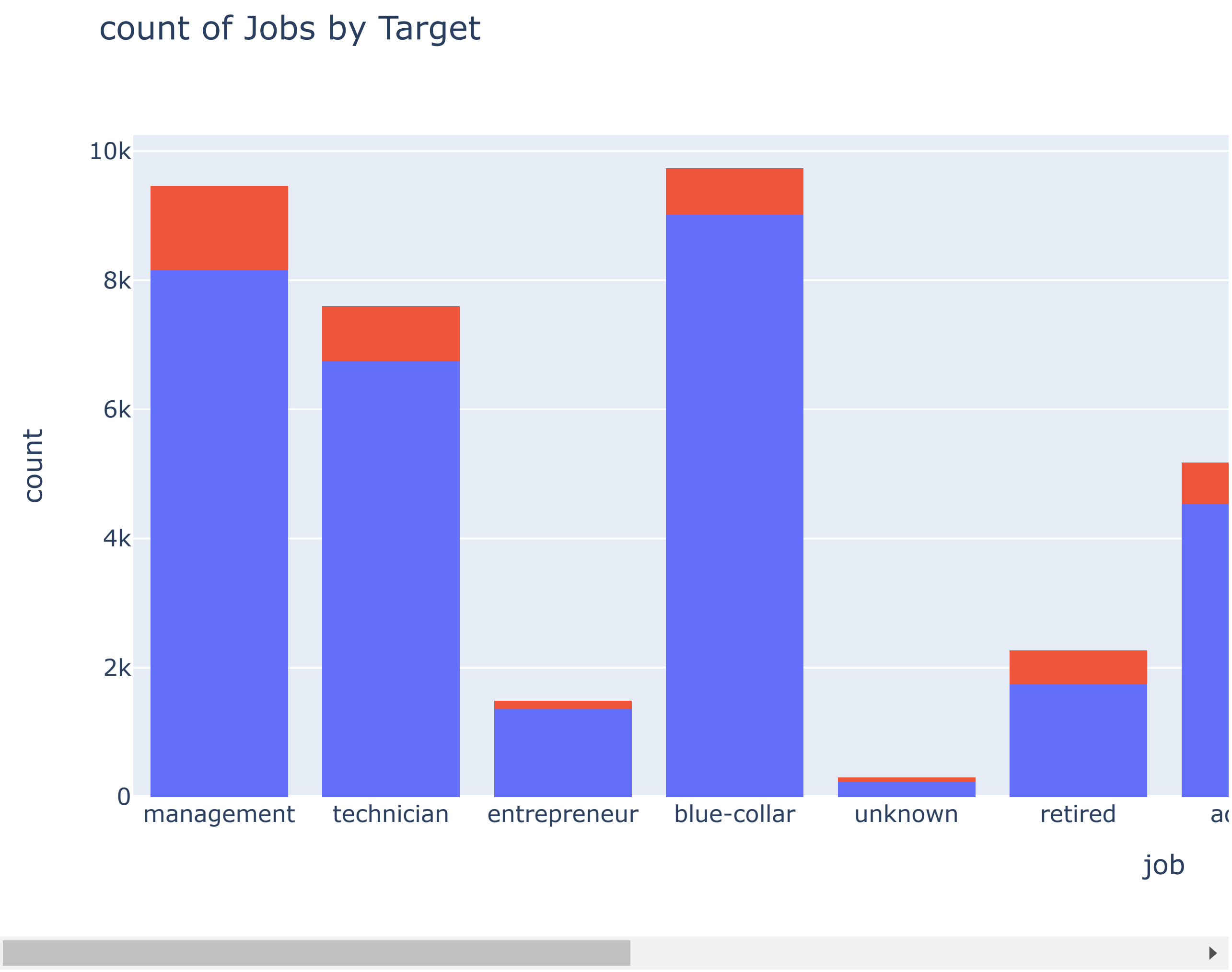
```
df["job"].value_counts()
```

```
blue-collar      9732
management       9458
technician       7597
admin.           5171
services         4154
retired          2264
self-employed    1579
entrepreneur     1487
unemployed       1303
housemaid        1240
student           938
unknown           288
Name: job, dtype: int64
```

```
df.groupby("job")["target"].agg(sum)
```

```
job
admin.            631
blue-collar       708
entrepreneur      123
housemaid         109
management       1301
retired           516
self-employed     187
services          369
student           269
technician        840
unemployed        202
unknown            34
Name: target, dtype: int64
```

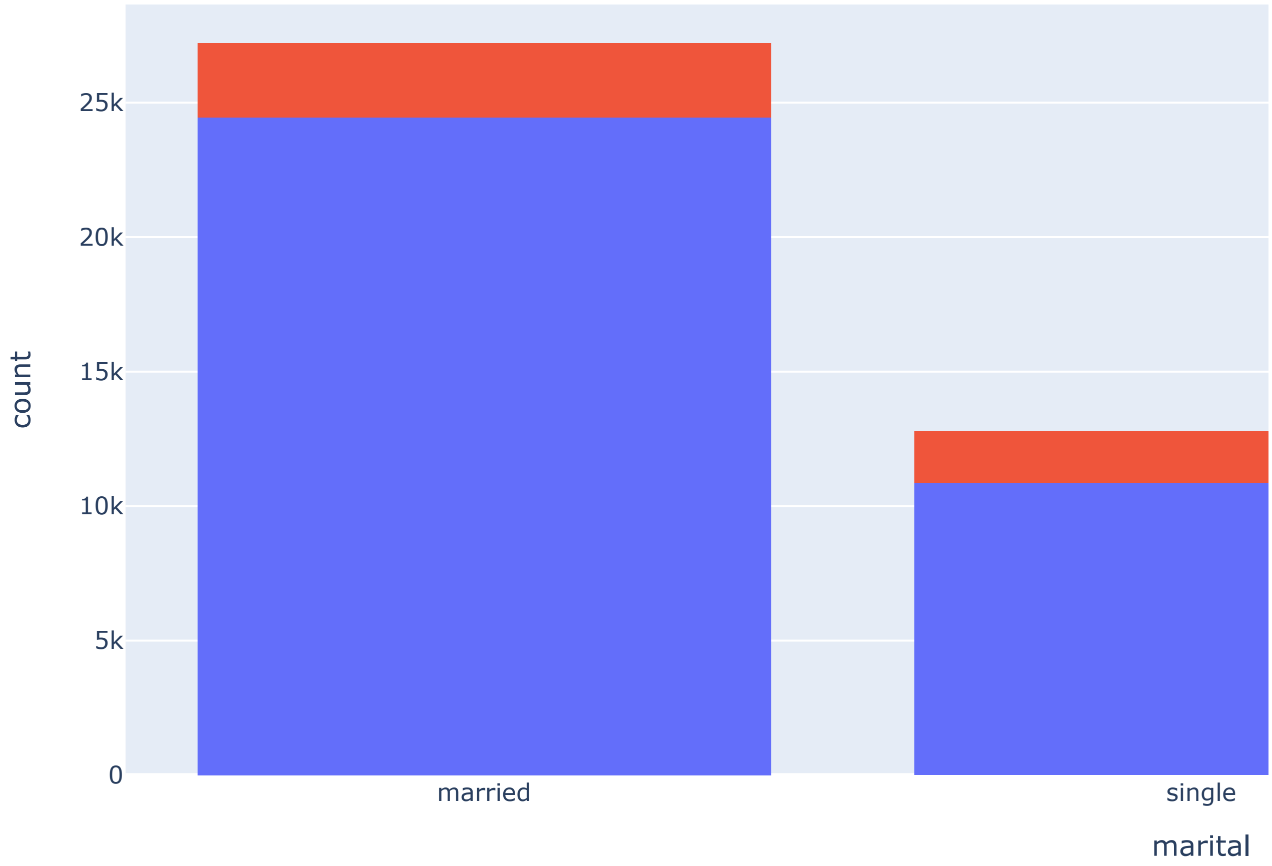
```
px.histogram(df,x="job",color="target",title='count of Jobs by Target',)
```



```
df["marital"].value_counts()
```

```
married     27214
single      12790
divorced     5207
Name: marital, dtype: int64
```

```
px.histogram(df,x="marital",color="target")
```



```
df.groupby("marital")["target"].agg(sum)
```

```
marital
divorced     622
married     2755
single      1912
Name: target, dtype: int64
```

```
df["education"].value_counts()
```

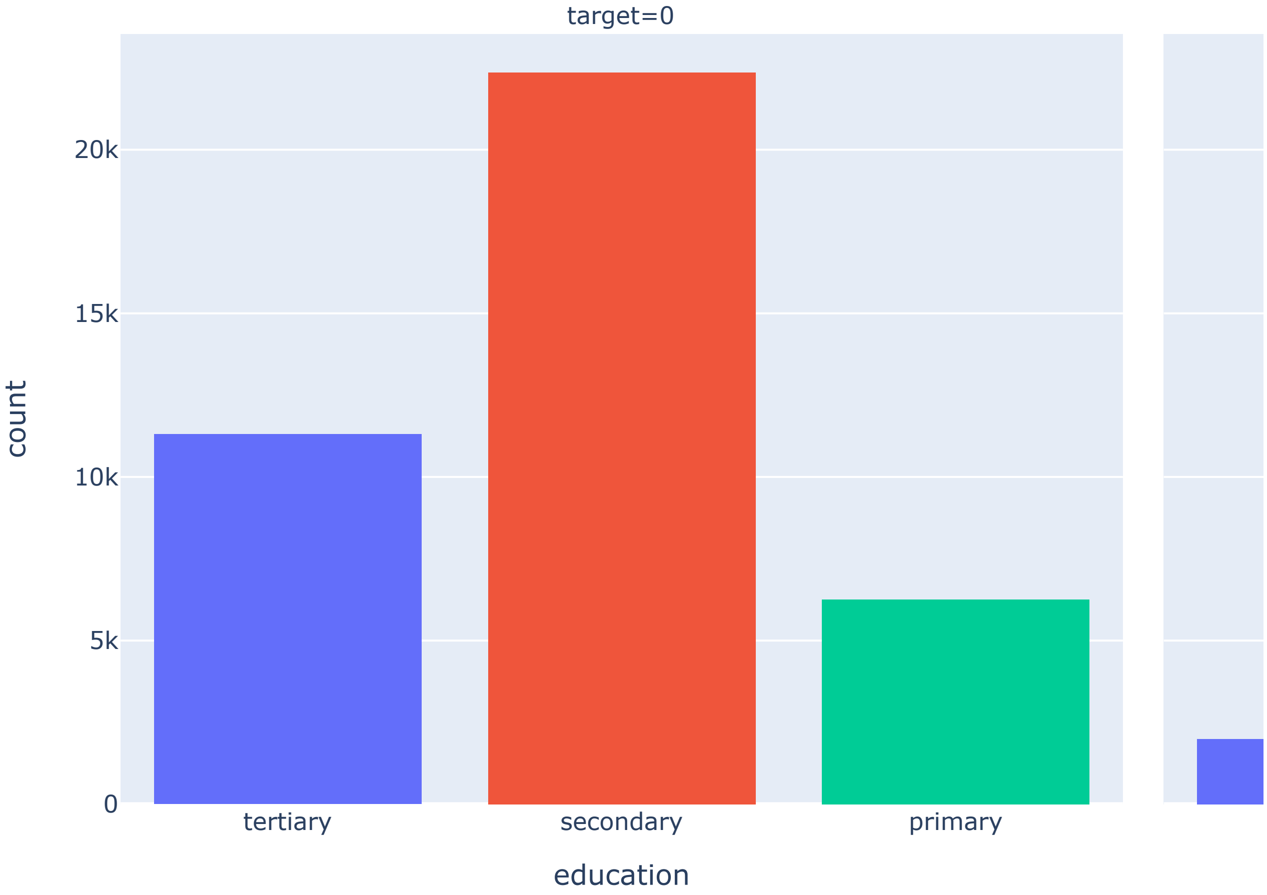
```
secondary    23202
tertiary     13301
primary       6851
unknown       1857
Name: education, dtype: int64
```

```
df["education"].replace("unknown",df["education"].mode()[0],inplace=True)
```

```
df["education"].value_counts()
```

```
secondary    25059
tertiary     13301
primary       6851
Name: education, dtype: int64
```

```
px.histogram(df,x="education",color="education",facet_col="target")
```



```
df.default.value_counts()
```

```
no     44396
yes      815
Name: default, dtype: int64
```

```
df.groupby("default")["target"].agg(sum)
```

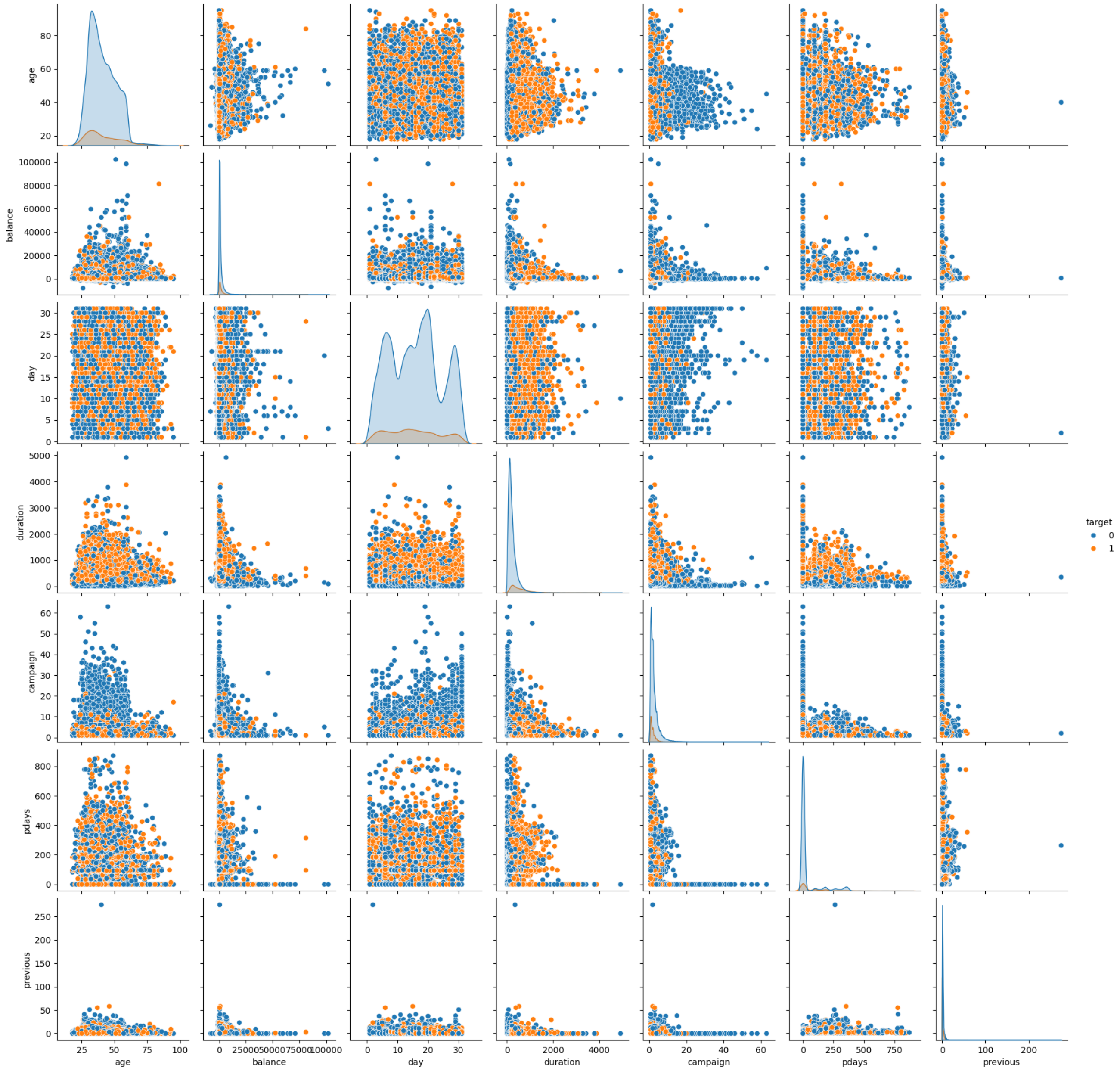
```
default
no     5237
yes      52
Name: target, dtype: int64
```

```
num=df.loc[:,['age','balance', 'day', 'duration', 'campaign','loan', 'pdays','previous','
```

```
?sns.pairplot
```

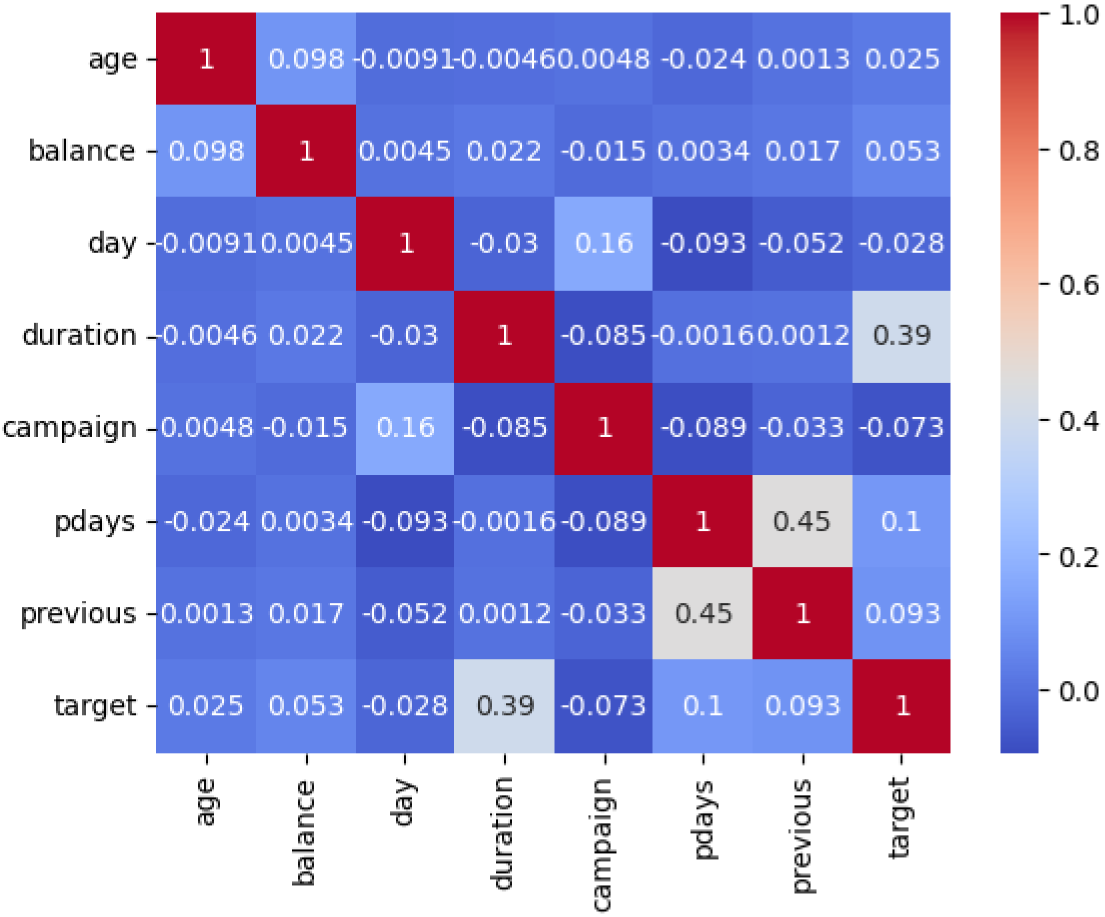
```
sns.pairplot(num,hue="target")
```

```
<seaborn.axisgrid.PairGrid at 0x7cb80327bfd0>
```

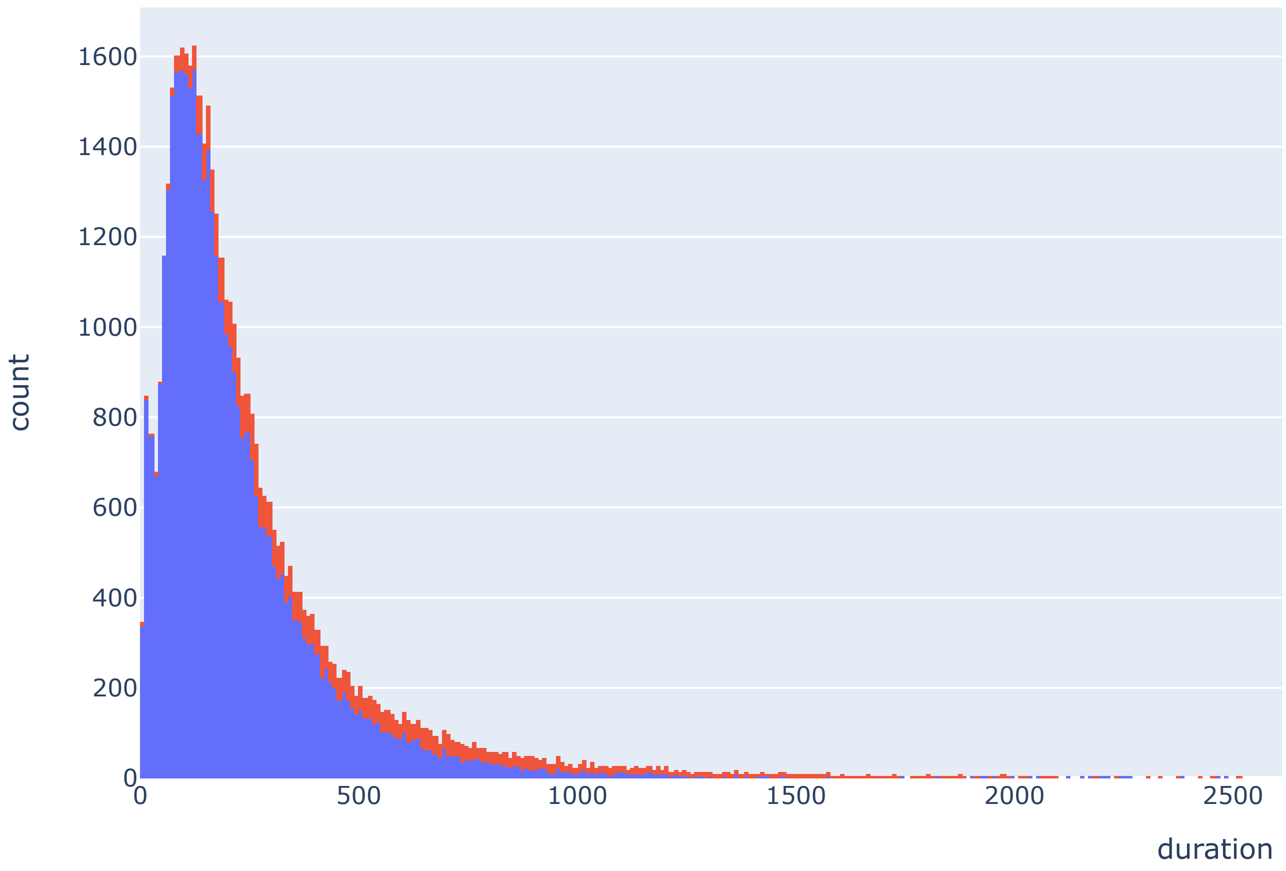

age
balance
day
duration
campaign
pdays
previous
target
0
1
80
60
40
20
100000
80000
60000
40000
20000
0
30
25
20
15
10
5
0
5000
4000
3000
2000
1000
0
60
50
40
30
20
10
0
800
600
400
200
0
250
200
150
100
50
0
25 50 75 100
0 250000500000750001000000
0 10 20 30
0 2000 4000
0 20 40 60
0 250 500 750
0 100 200

```
cm=num.corr()
sns.heatmap(cm,annot=True,cmap="coolwarm")
```

```
<Axes: >
```

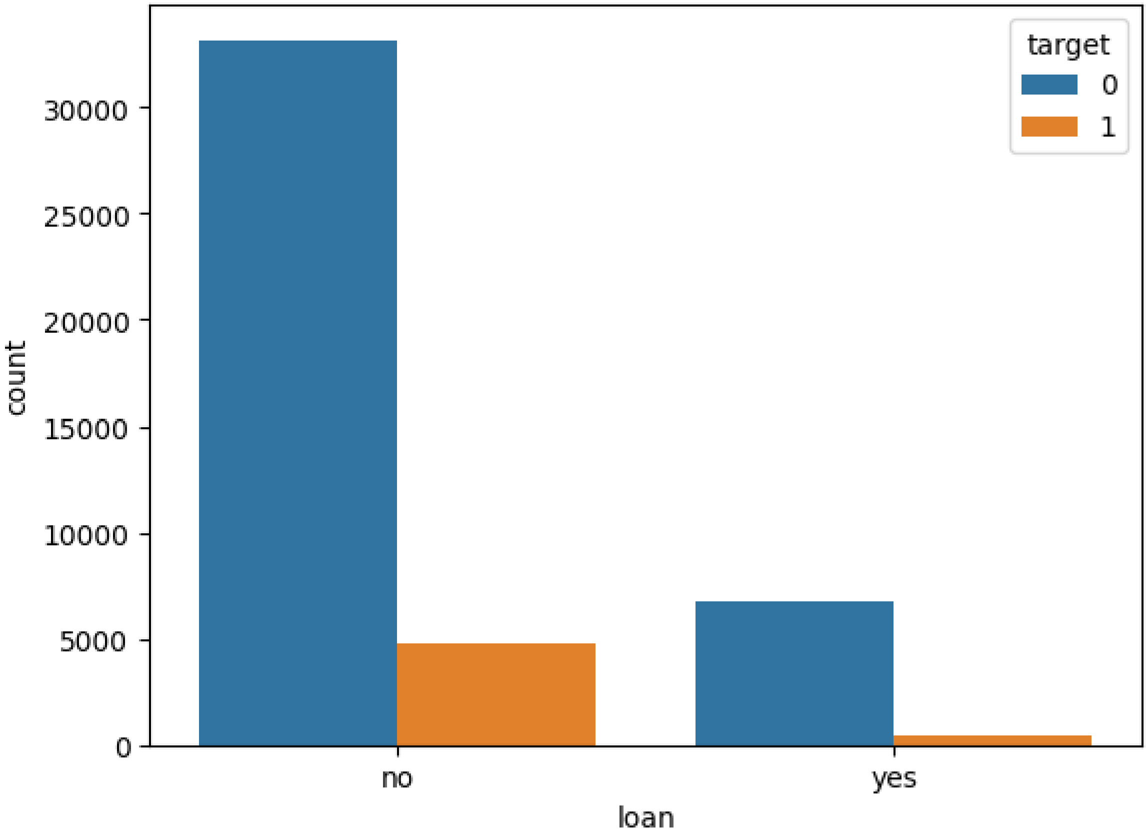


```
px.histogram(df,x="duration",color="target")
```



```
sns.countplot(data=df,x="loan",hue="target")
```

```
<Axes: xlabel='loan', ylabel='count'>
```



Based on these correlations, variables such as duration, pdays, and previous seem to have relatively stronger correlations with the target variable compared to others.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 18 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   age        45211 non-null  int64 
 1   job        45211 non-null  object
 2   marital    45211 non-null  object
 3   education  45211 non-null  object
 4   default    45211 non-null  object
 5   balance    45211 non-null  int64 
 6   housing    45211 non-null  object
 7   loan       45211 non-null  object
 8   contact    45211 non-null  object
 9   day        45211 non-null  int64 
 10  month      45211 non-null  object
 11  duration   45211 non-null  int64 
 12  campaign   45211 non-null  int64 
 13  pdays      45211 non-null  int64 
 14  previous   45211 non-null  int64 
 15  poutcome   45211 non-null  object
 16  y          45211 non-null  object
 17  target     45211 non-null  int64 
dtypes: int64(8), object(10)
memory usage: 6.2+ MB
```

```
cat=df.loc[:,['job', 'marital', 'education', 'default', 'housing','loan', 'contact', 'mon
cat.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 10 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   job        45211 non-null  object
 1   marital    45211 non-null  object
 2   education  45211 non-null  object
 3   default    45211 non-null  object
 4   housing    45211 non-null  object
 5   loan       45211 non-null  object
 6   contact    45211 non-null  object
 7   month      45211 non-null  object
 8   poutcome   45211 non-null  object
 9   y          45211 non-null  object
dtypes: object(10)
memory usage: 3.4+ MB
```

```
categorical_columns = ["job", "marital", "education", "default", "housing", "loan", "cont
target_variable = "y"


association_results = pd.DataFrame(columns=["Column", "Chi-square", "P-value"])


for column in categorical_columns:
    contingency_table = pd.crosstab(df[column], df[target_variable])
    chi2, p, _, _ = chi2_contingency(contingency_table)
    association_results = association_results.append({"Column": column, "Chi-square": chi


print(association_results)
```

```
      Column   Chi-square        P-value
0        job   836.105488  3.337122e-172
1    marital   196.495946   2.145100e-43
2  education   223.834823   2.482480e-49
3    default    22.202250   2.453861e-06
4    housing   874.822449  2.918798e-192
5       loan   209.616980   1.665061e-47
6    contact  1035.714225  1.251738e-225
7      month  3061.838938   0.000000e+00
8   poutcome  4391.506589   0.000000e+00
```

```
x=df[['job', 'marital', 'education', 'default', 'housing','loan', 'contact', 'month','pou
```

```
dq=df.copy()
```

```
x.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 14 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   job        45211 non-null  object
 1   marital    45211 non-null  object
 2   education  45211 non-null  object
 3   default    45211 non-null  object
 4   housing    45211 non-null  object
 5   loan       45211 non-null  object
 6   contact    45211 non-null  object
 7   month      45211 non-null  object
 8   poutcome   45211 non-null  object
 9   duration   45211 non-null  int64 
 10  pdays      45211 non-null  int64 
 11  previous   45211 non-null  int64 
 12  campaign   45211 non-null  int64 
 13  balance    45211 non-null  int64 
dtypes: int64(5), object(9)
memory usage: 4.8+ MB
```

```
y=df["target"]
y.info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 45211 entries, 0 to 45210
Series name: target
Non-Null Count  Dtype
--------------  -----
45211 non-null  int64
dtypes: int64(1)
memory usage: 353.3 KB
```

```
x
```

	job	marital	education	default	housing	loan	contact	month	poutc
0	management	married	tertiary	no	yes	no	unknown	may	unkn
1	technician	single	secondary	no	yes	no	unknown	may	unkn
2	entrepreneur	married	secondary	no	yes	yes	unknown	may	unkn
3	blue-collar	married	secondary	no	yes	no	unknown	may	unkn
4	unknown	single	secondary	no	no	no	unknown	may	unkn
...	...	...	...	...	...	...	...	...	
45206	technician	married	tertiary	no	no	no	cellular	nov	unkn
45207	retired	divorced	primary	no	no	no	cellular	nov	unkn
45208	retired	married	secondary	no	no	no	cellular	nov	succ
45209	blue-collar	married	secondary	no	no	no	telephone	nov	unkn
45210	entrepreneur	married	secondary	no	no	no	cellular	nov	c

45211 rows × 14 columns

```
x_org=x.copy()
```

```
x_org
```

	job	marital	education	default	housing	loan	contact	month	poutc
0	management	married	tertiary	no	yes	no	unknown	may	unkn
1	technician	single	secondary	no	yes	no	unknown	may	unkn
2	entrepreneur	married	secondary	no	yes	yes	unknown	may	unkn
3	blue-collar	married	secondary	no	yes	no	unknown	may	unkn
4	unknown	single	secondary	no	no	no	unknown	may	unkn
...	...	...	...	...	...	...	...	...	
45206	technician	married	tertiary	no	no	no	cellular	nov	unkn
45207	retired	divorced	primary	no	no	no	cellular	nov	unkn
45208	retired	married	secondary	no	no	no	cellular	nov	succ
45209	blue-collar	married	secondary	no	no	no	telephone	nov	unkn
45210	entrepreneur	married	secondary	no	no	no	cellular	nov	c

45211 rows × 14 columns

```
x1=pd.get_dummies(x_org)
x1
```

	duration	pdays	previous	campaign	balance	job_admin.	job_blue-collar	job_entre
0	261	-1	0	1	2143	0	0	
1	151	-1	0	1	29	0	0	
2	76	-1	0	1	2	0	0	
3	92	-1	0	1	1506	0	1	
4	198	-1	0	1	1	0	0	
...	...	...	...	...	...	...	...	
45206	977	-1	0	3	825	0	0	
45207	456	-1	0	2	1729	0	0	
45208	1127	184	3	5	5715	0	0	
45209	508	-1	0	4	668	0	1	
45210	361	188	11	2	2971	0	0	

45211 rows × 48 columns

```
from sklearn.model_selection  import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x1,y,test_size=0.2,random_state=42)
```

```
x_train.shape,y_train.shape
```

```
((36168, 48), (36168,))
```

```
x_test.shape,y_test.shape
```

```
((9043, 48), (9043,))
```

```




dt=DecisionTreeClassifier()
dt.fit(x_train,y_train)

y_train_pred = dt.predict(x_train)

train_acc=accuracy_score(y_train,y_train_pred)
```