In [1]:
```
import pandas as pd
import seaborn as sns
import sklearn
```

In [4]:
```
df = pd.read_csv('../data/bank/bank.csv', sep=';')
df.head()
```

Out[4]:

	age	job	marital	education	default	balance	housing	loan	contact	day	month
0	30	unemployed	married	primary	no	1787	no	no	cellular	19	oc
1	33	services	married	secondary	no	4789	yes	yes	cellular	11	ma
2	35	management	single	tertiary	no	1350	yes	no	cellular	16	ap
3	30	management	married	tertiary	no	1476	yes	yes	unknown	3	ju
4	59	blue-collar	married	secondary	no	0	yes	no	unknown	5	ma

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4521 entries, 0 to 4520
Data columns (total 17 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   age        4521 non-null   int64 
 1   job        4521 non-null   object
 2   marital    4521 non-null   object
 3   education  4521 non-null   object
 4   default    4521 non-null   object
 5   balance    4521 non-null   int64 
 6   housing    4521 non-null   object
 7   loan       4521 non-null   object
 8   contact    4521 non-null   object
 9   day        4521 non-null   int64 
 10  month      4521 non-null   object
 11  duration   4521 non-null   int64 
 12  campaign   4521 non-null   int64 
 13  pdays      4521 non-null   int64 
 14  previous   4521 non-null   int64 
 15  poutcome   4521 non-null   object
 16  y          4521 non-null   object
dtypes: int64(7), object(10)
memory usage: 600.6+ KB
```

In [7]:
```
df.describe()
```

Out[7]:

	age	balance	day	duration	campaign	pdays	
count	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000	4521
mean	41.170095	1422.657819	15.915284	263.961292	2.793630	39.766645	
std	10.576211	3009.638142	8.247667	259.856633	3.109807	100.121124	
min	19.000000	-3313.000000	1.000000	4.000000	1.000000	-1.000000	
25%	33.000000	69.000000	9.000000	104.000000	1.000000	-1.000000	
50%	39.000000	444.000000	16.000000	185.000000	2.000000	-1.000000	
75%	49.000000	1480.000000	21.000000	329.000000	3.000000	-1.000000	
max	87.000000	71188.000000	31.000000	3025.000000	50.000000	871.000000	25

In [12]:
```
from scipy import stats
import numpy as np
```

In [38]:
```
a = np.abs(stats.zscore(df['balance']))<float(2)
```

In [41]:
```
df[(np.abs(stats.zscore(df['balance']))<float(2))]
```

Out[41]:

	age	job	marital	education	default	balance	housing	loan	contact	day	m
0	30	unemployed	married	primary	no	1787	no	no	cellular	19	
1	33	services	married	secondary	no	4789	yes	yes	cellular	11	
2	35	management	single	tertiary	no	1350	yes	no	cellular	16	
3	30	management	married	tertiary	no	1476	yes	yes	unknown	3	
4	59	blue-collar	married	secondary	no	0	yes	no	unknown	5	
...	...	...	...	...	...	...	...	...	...	...	
4516	33	services	married	secondary	no	-333	yes	no	cellular	30	
4517	57	self-employed	married	tertiary	yes	-3313	yes	yes	unknown	9	
4518	57	technician	married	secondary	no	295	no	no	cellular	19	
4519	28	blue-collar	married	secondary	no	1137	no	no	cellular	6	
4520	44	entrepreneur	single	tertiary	no	1136	yes	yes	cellular	3	

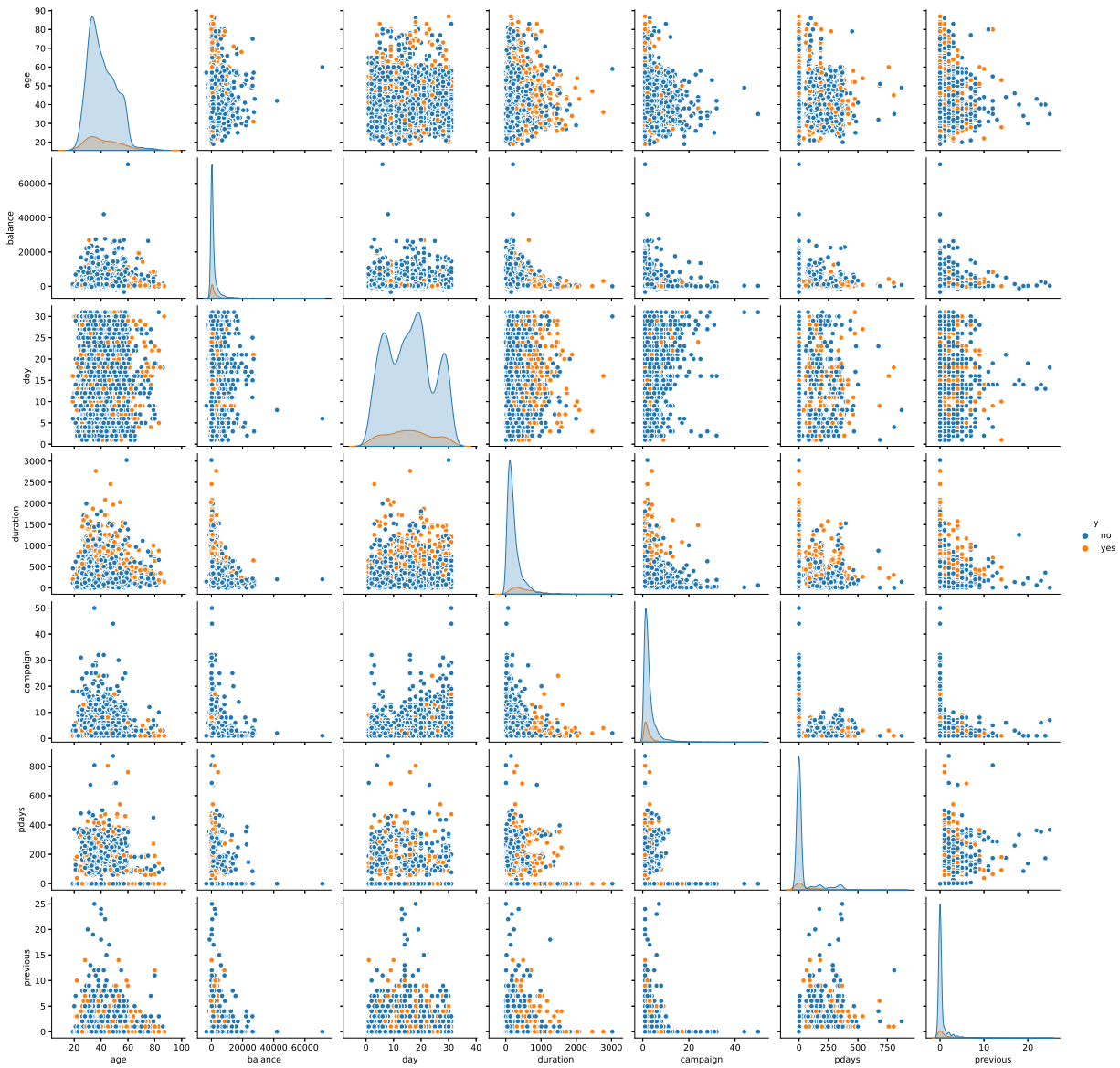
4355 rows × 17 columns

In [53]:
```
df.groupby('job')['y'].value_counts()
```

Out[53]:
```
job             y
admin.          no     420
                yes     58
blue-collar     no     877
                yes     69
entrepreneur    no     153
                yes     15
housemaid       no      98
                yes     14
management      no     838
                yes    131
retired         no     176
                yes     54
self-employed   no     163
                yes     20
services        no     379
                yes     38
student         no      65
                yes     19
technician      no     685
                yes     83
unemployed      no     115
                yes     13
unknown         no      31
                yes      7
Name: y, dtype: int64
```

In [46]:
```
sns.pairplot(data=df,hue = 'y')
```

Out[46]: <seaborn.axisgrid.PairGrid at 0x7fd96e537d90>



In [44]:
```
condition = np.array([True]*df.shape[0])
for col in df.columns:
    if df[col].dtype == 'int64':
        condition &= (np.abs(stats.zscore(df[col]))<float(3))
```

```
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
(4521,)
```