Import Libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

import warnings
warnings.filterwarnings('ignore')

import plotly
import plotly.graph_objs as go
from plotly.offline import download_plotlyjs,init_notebook_mode, iplot , pl
init_notebook_mode(connected= True)
```

```
C:\Users\loves\AppData\Roaming\Python\Python310\site-packages\pandas\core
\arrays\masked.py:60: UserWarning: Pandas requires version '1.3.6' or newe
r of 'bottleneck' (version '1.3.5' currently installed).
  from pandas.core import (
```

Load The Dataset

In [2]:
```
df = pd.read_csv("bank-additional.csv", sep =';')

df.head(5)
```

Out[2]:

	age	job	marital	education	default	housing	loan	contact	month	day_
0	30	blue-collar	married	basic.9y	no	yes	no	cellular	may	
1	39	services	single	high.school	no	no	no	telephone	may	
2	25	services	married	high.school	no	yes	no	telephone	jun	
3	38	services	married	basic.9y	no	unknown	unknown	telephone	jun	
4	47	admin.	married	university.degree	no	yes	no	cellular	nov	

5 rows × 21 columns

In [3]:
```
d ={'no': 0, 'yes': 1}

df['y'] = df['y'].map(d)
```

Attribute Information

In [4]:
```
df.columns
```

Out[4]:
```
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', 'y'],
      dtype='object')
```

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

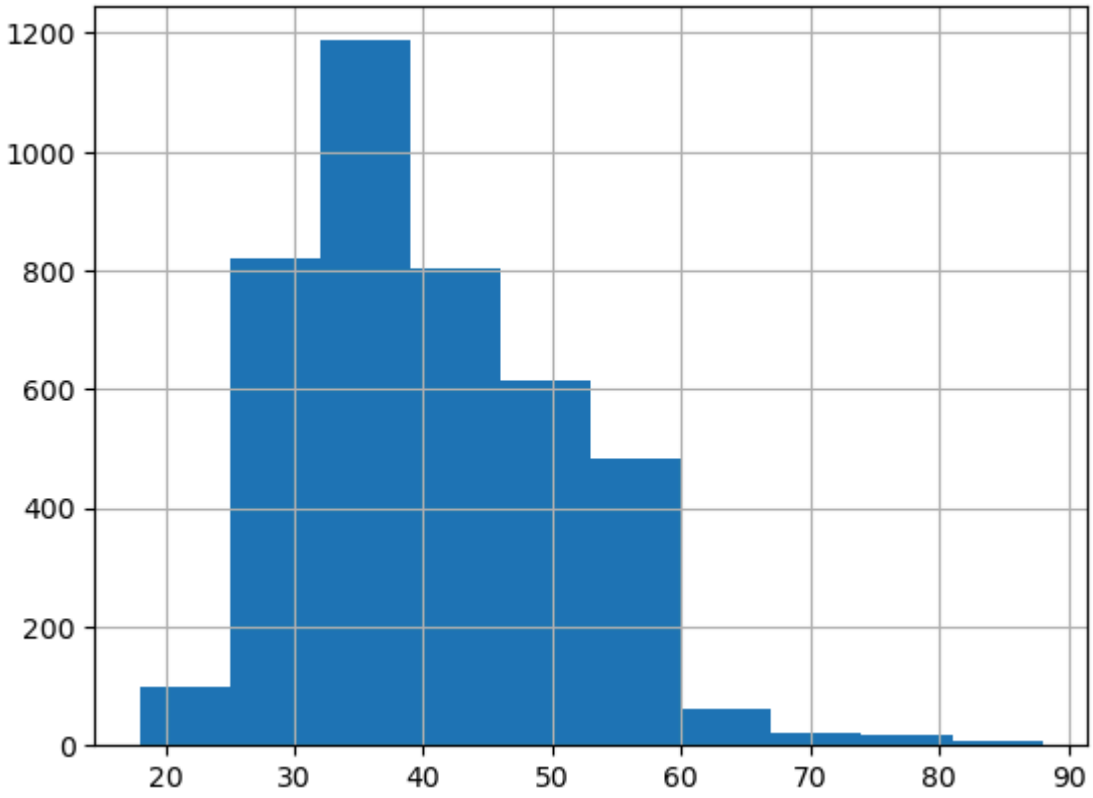
Out[5]:
```
(4119, 21)
```

Overview of Pythojn libraries for visual data analysis

Matplotlib

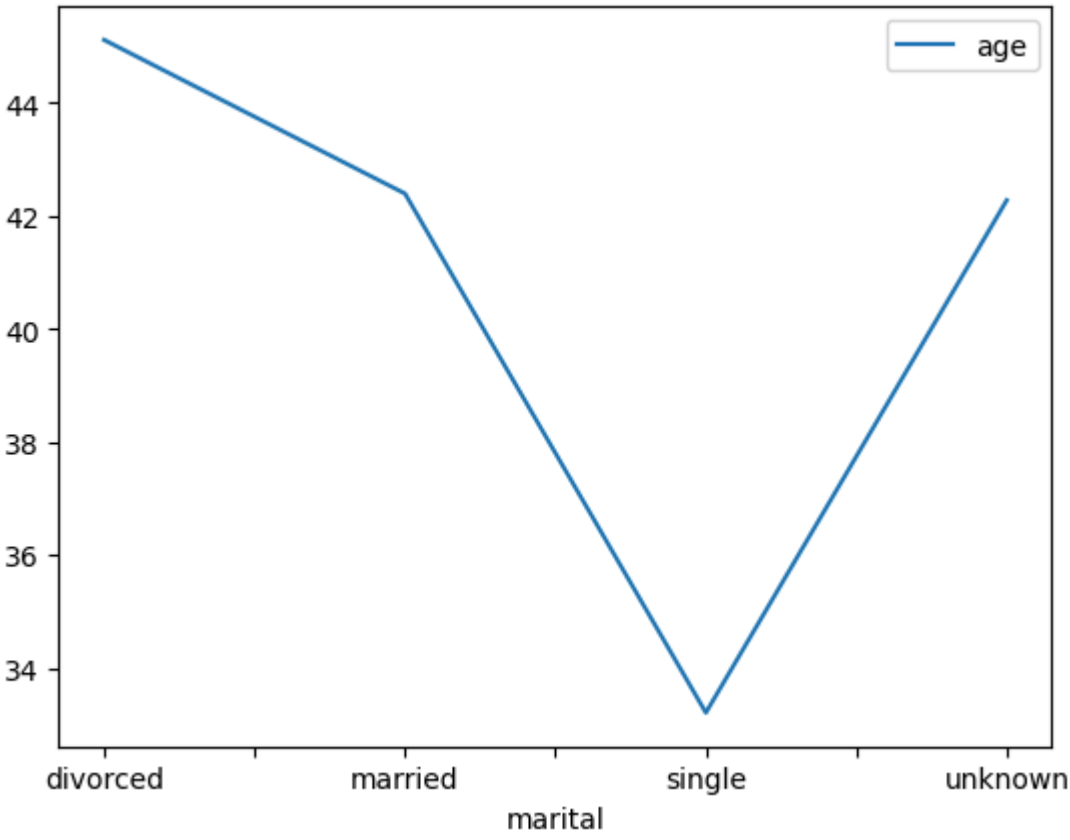
In [6]:
```
df['age'].hist()
```

Out[6]:
```
<Axes: >
```

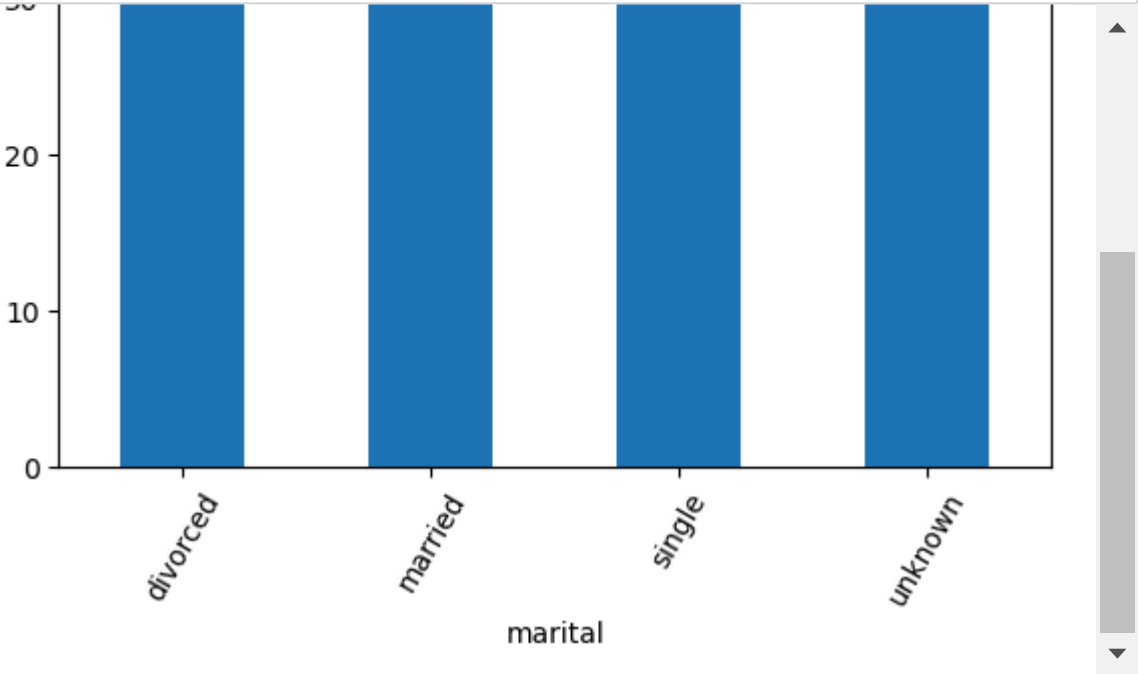


In [7]:
```
df[["age","marital"]].groupby("marital").mean().plot()
```

Out[7]:
```
<Axes: xlabel='marital'>
```



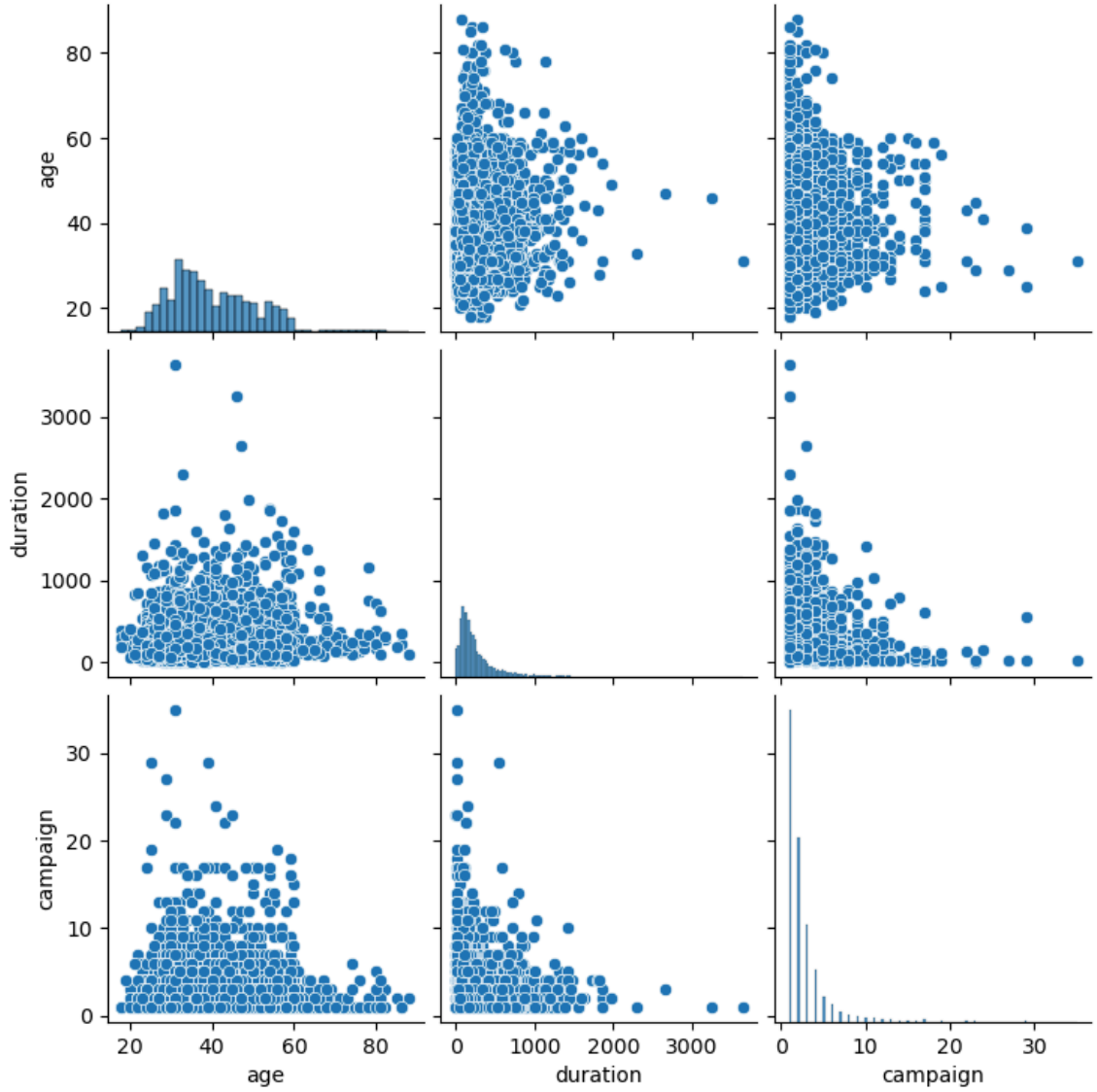
In [8]:
```
df[["age","marital"]].groupby("marital").mean().plot(kind='bar', rot =60)
```



Seaborn

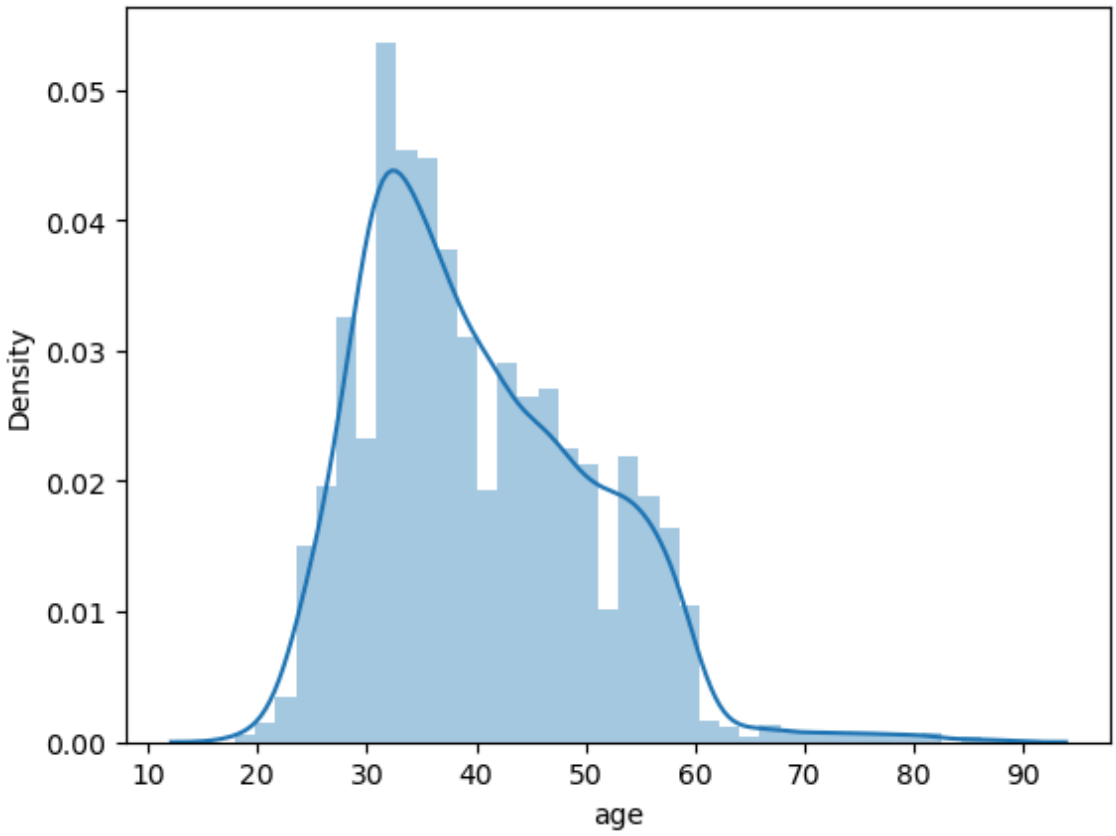
In [9]:
```
sns.pairplot(df[["age","duration","campaign"]])
```

Out[9]:
```
<seaborn.axisgrid.PairGrid at 0x21455e784f0>
```



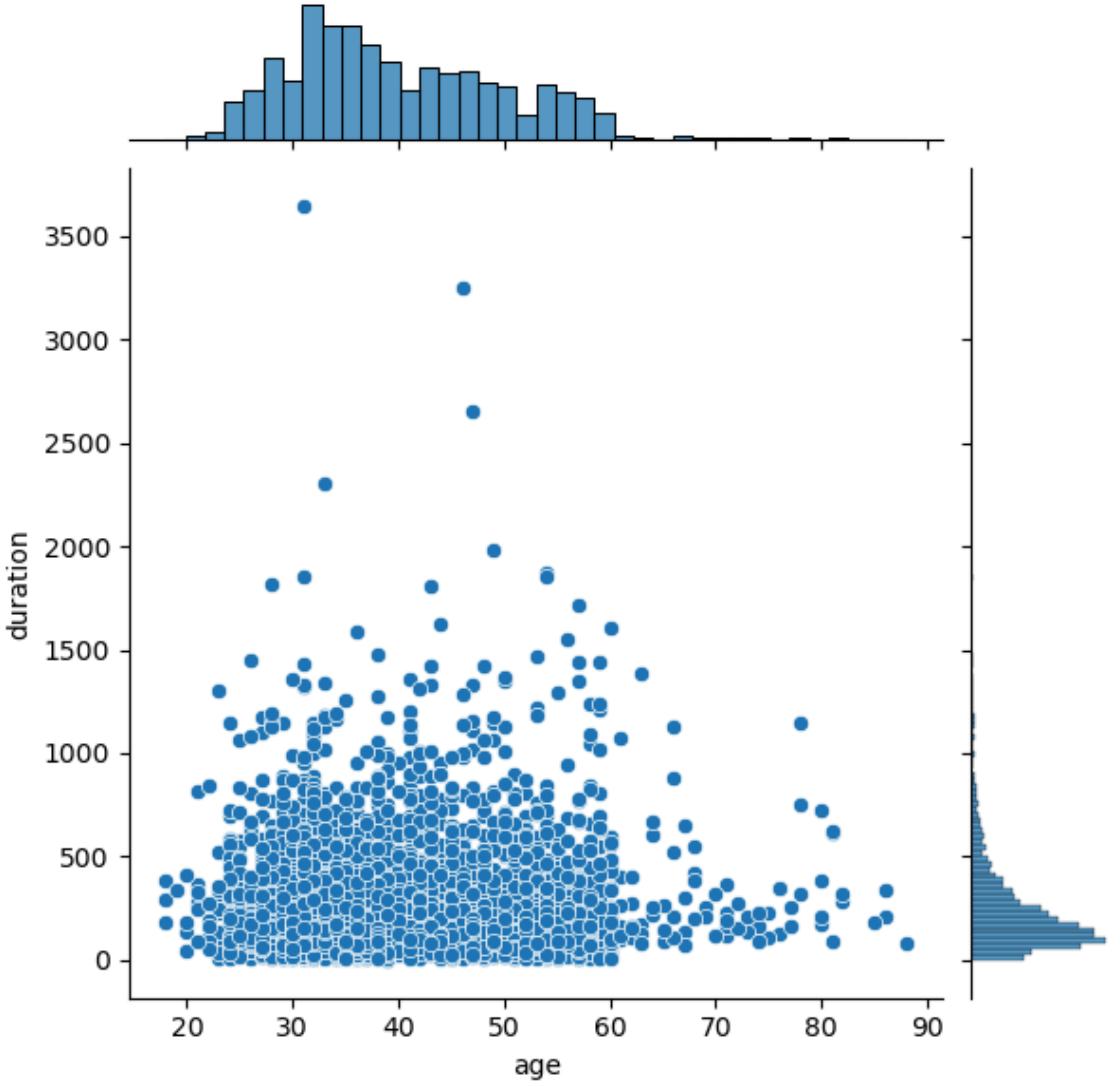
In [10]:
```
sns.distplot(df["age"])

# displays histogram and kernal density estimation
```

Out[10]: <Axes: xlabel='age', ylabel='Density'>



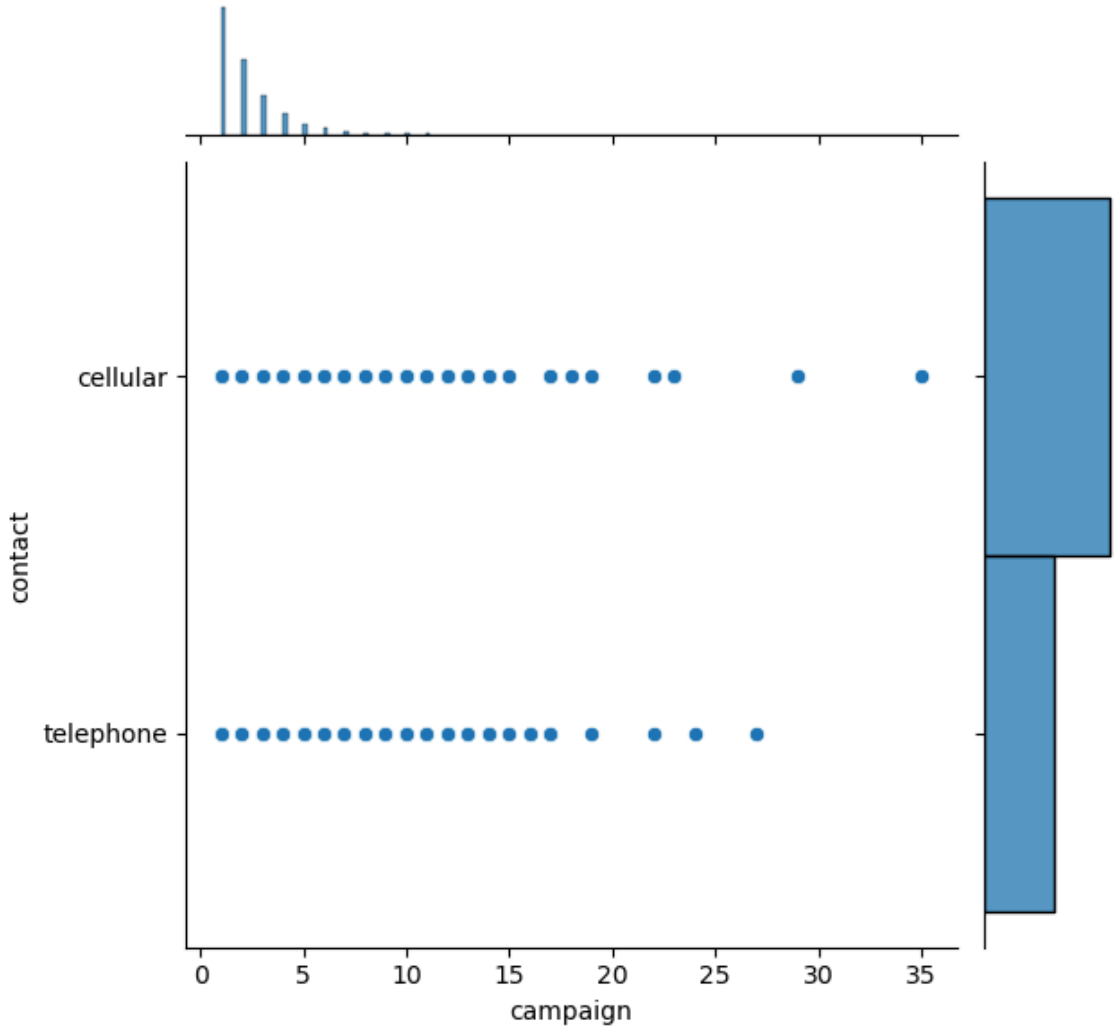
In [11]:
```
sns.jointplot(x="age",y="duration",data=df, kind="scatter")

## joint_plot this is a hybrid scatterplot and histograms
```

Out[11]: <seaborn.axisgrid.JointGrid at 0x2145789ed40>



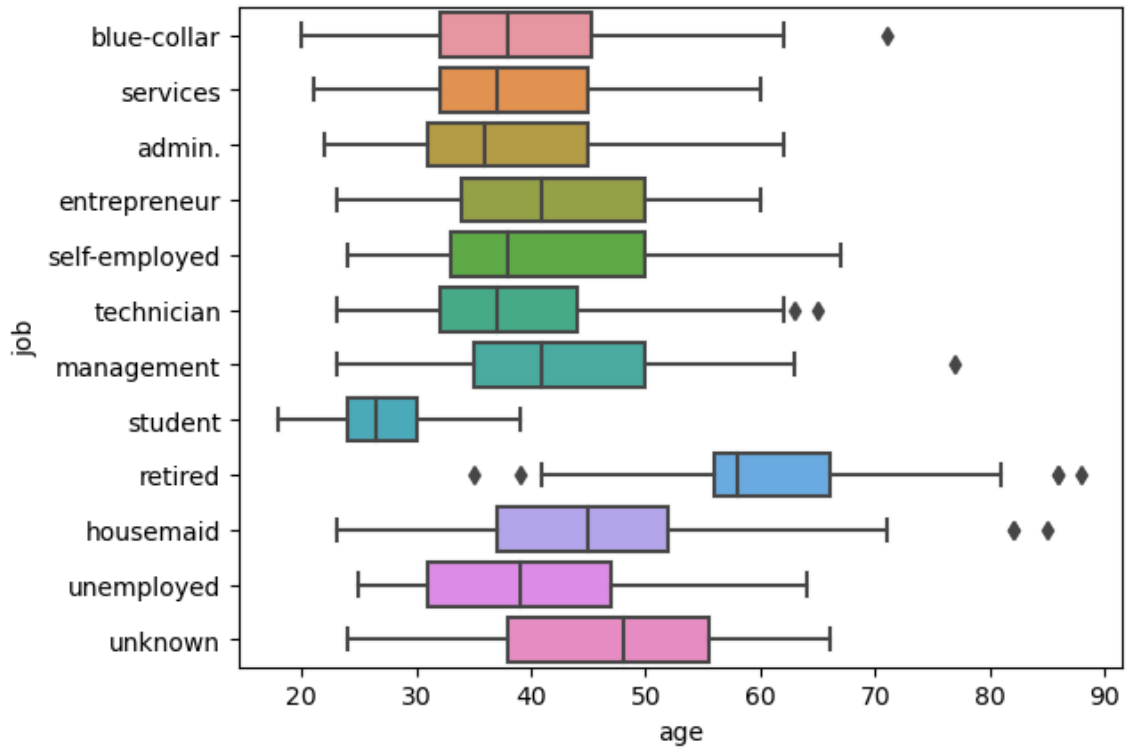
In [12]:
```
sns.jointplot(x="campaign", y= "contact",kind='scatter',data=df)
```

Out[12]:
```
<seaborn.axisgrid.JointGrid at 0x214590ee4d0>
```



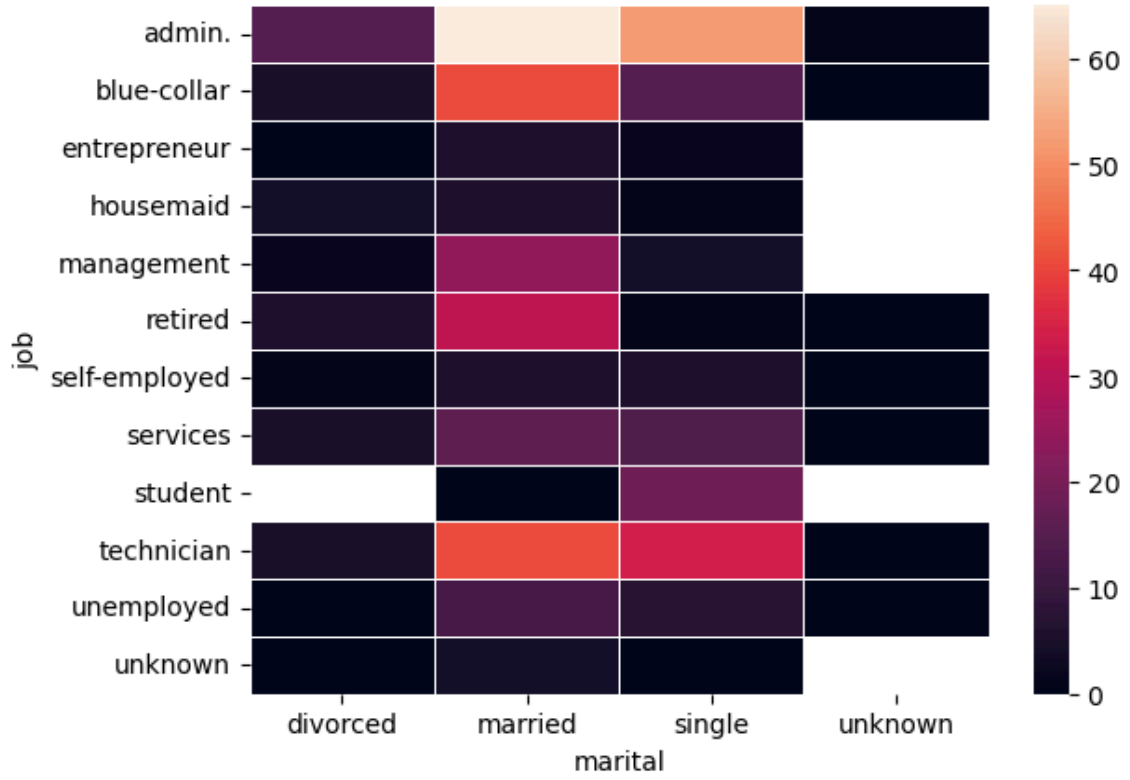
In [13]:
```
sns.boxplot(y="job",x="age",data= df,orient='h')
```

Out[13]:
```
<Axes: xlabel='age', ylabel='job'>
```



In [14]:
```
job_marital_y= (
   df.pivot_table(
         index="job", columns="marital", values="y", aggfunc=sum
   )
)
sns.heatmap(job_marital_y, annot=True, fmt="d", linewidths=0.5);
```

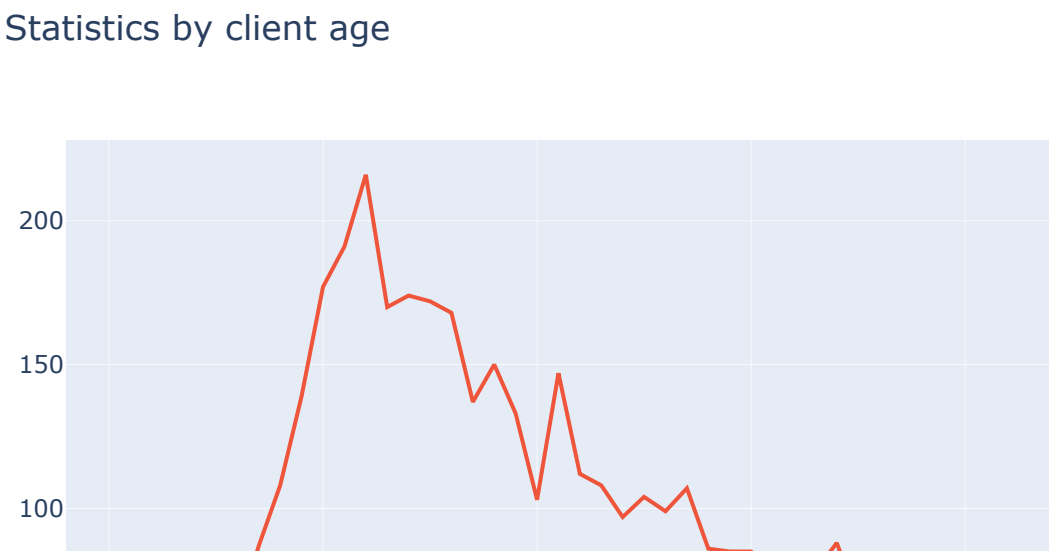
```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[14], line 6
      1 job_marital_y= (
      2    df.pivot_table(
      3          index="job", columns="marital", values="y", aggfunc=sum
      4    )
      5 )
----> 6 sns.heatmap(job_marital_y, annot=True, fmt="d", linewidths=0.5)

File C:\ProgramData\anaconda3\lib\site-packages\seaborn\matrix.py:459, in heatmap(data, vmin, vmax, cmap, center, robust, annot, fmt, annot_kws, linewidths, linecolor, cbar, cbar_kws, cbar_ax, square, xticklabels, yticklabels, mask, ax, **kwargs)
    457 if square:
    458     ax.set_aspect("equal")
--> 459 plotter.plot(ax, cbar_ax, kwargs)
    460 return ax

File C:\ProgramData\anaconda3\lib\site-packages\seaborn\matrix.py:352, in _HeatMapper.plot(self, ax, cax, kws)
    350 # Annotate the cells with the formatted values
    351 if self.annot:
--> 352     self._annotate_heatmap(ax, mesh)

File C:\ProgramData\anaconda3\lib\site-packages\seaborn\matrix.py:260, in _HeatMapper._annotate_heatmap(self, ax, mesh)
    258 lum = relative_luminance(color)
    259 text_color = ".15" if lum > .408 else "w"
--> 260 annotation = ("{:" + self.fmt + "}").format(val)
    261 text_kwargs = dict(color=text_color, ha="center", va="center")
    262 text_kwargs.update(self.annot_kws)

ValueError: Unknown format code 'd' for object of type 'float'
```



Plotly

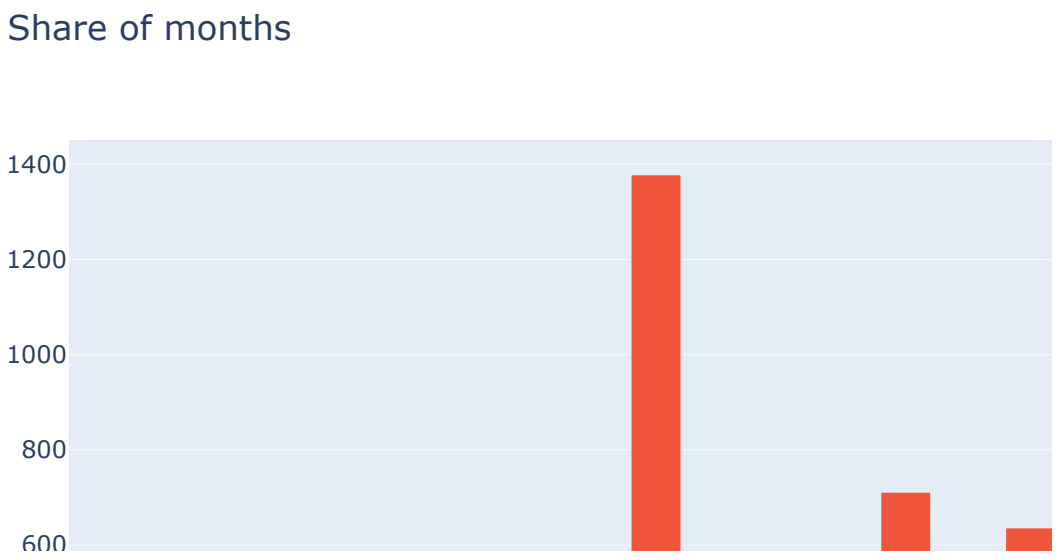
In [15]:
```
age_df= (
df.groupby("age")[["y"]].sum().join(df.groupby("age")[["y"]].count(), rsuff
)

age_df.columns=["Attracted","Total Number"]
```

In [16]:
```
trace0 = go.Scatter(x=age_df.index, y=age_df["Attracted"], name="Attracted"
trace1 = go.Scatter(x=age_df.index, y=age_df["Total Number"], name="Total N

data = [trace0, trace1]
layout = {"title": "Statistics by client age"}

fig=go.Figure(data=data, layout=layout)

iplot(fig, show_link=False)
```

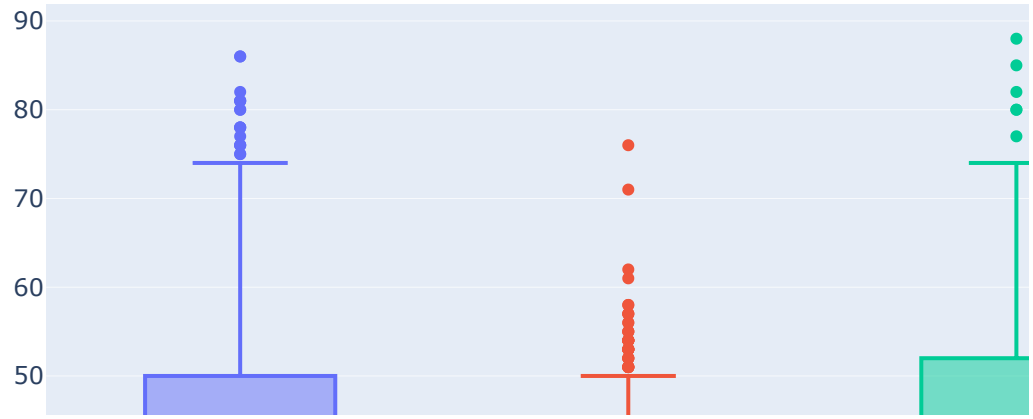


In [17]:
```
month_index = ['jan','feb','mar','apr','may','jun','jul','aug','sep','oct',
month_df = (
    df.groupby("month")[["y"]].sum().join(df.groupby("month")[["y"]].count(
).reindex(month_index)

month_df.columns=["Attracted","Total Number"]
```

In [18]:
```
trace0 = go.Bar(x=month_df.index, y=month_df["Attracted"], name="Attracted"
trace1 = go.Bar(x=month_df.index, y=month_df["Total Number"], name="Total N

data = [trace0, trace1]
layout = {"title": "Share of months"}

fig = go.Figure(data=data, layout=layout)

iplot(fig, show_link=False)
```



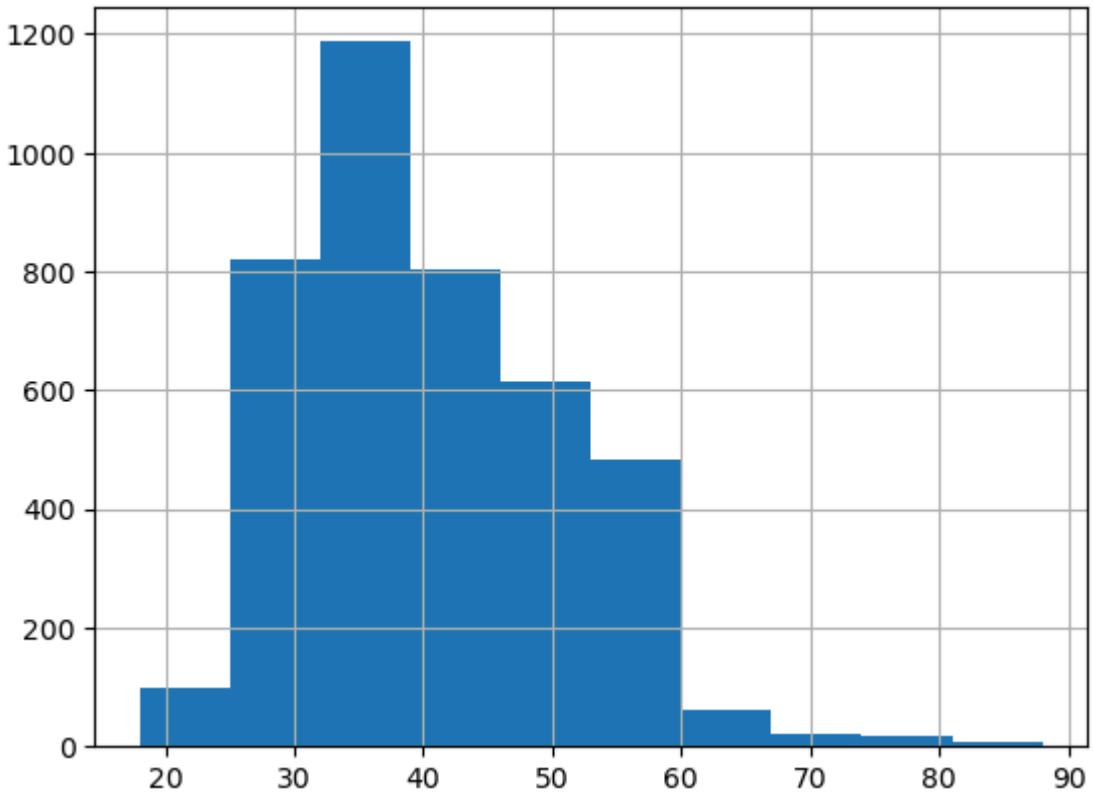
In [19]:
```
data = []

for status in df.marital.unique():
    data.append(go.Box(y=df[df.marital == status].age, name=status))
iplot(data, show_link=False)
```



Visual Analysis of Single features

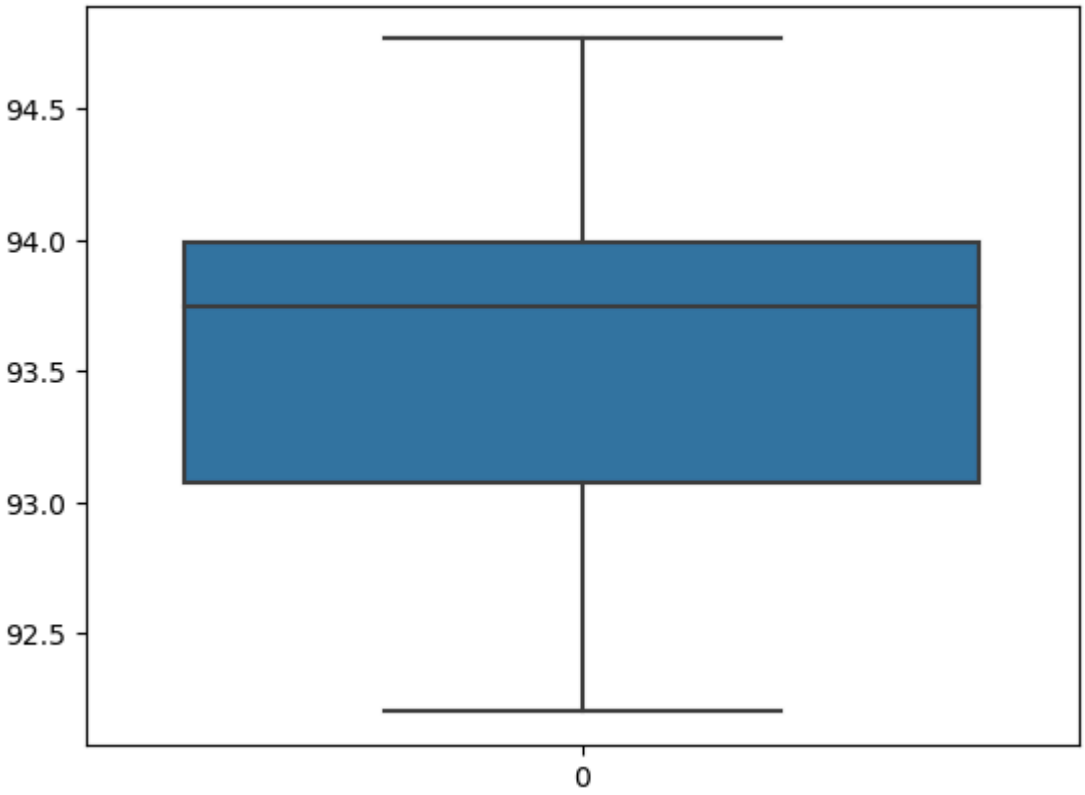
In [20]:
```
df["age"].hist()
```

Out[20]: <Axes: >



In [21]:
```
sns.boxplot(df["cons.price.idx"])
```

Out[21]:
```
<Axes: >
```



Categorical Features

Use the Countplot graphics for effective analysis of categorical features.It's effective to use the graphics of the type CountPlot for ananyzing categorical featurees

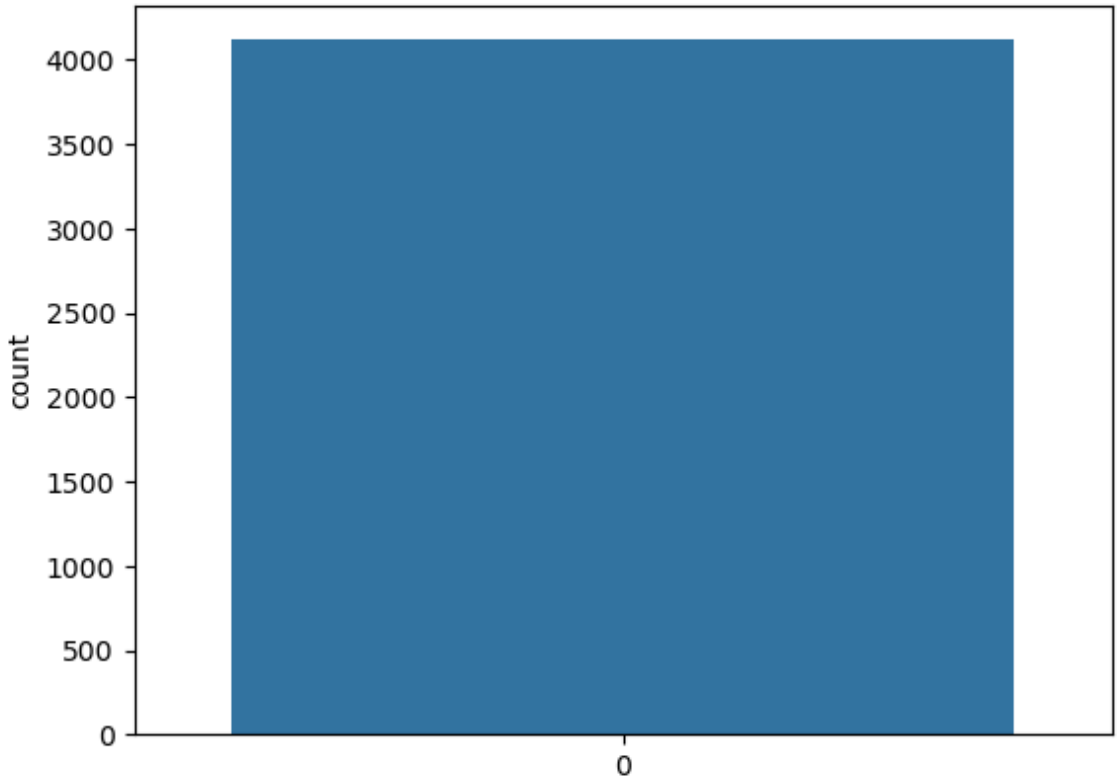
In [22]:
```
mar = df["marital"].value_counts().head()
```

In [23]:
```
df["y"].value_counts()
```

Out[23]:
```
y
0    3668
1     451
Name: count, dtype: int64
```

In [24]:
```
sns.countplot(df.y)
```

Out[24]:
```
<Axes: ylabel='count'>
```



In [25]:
```
#sns.countplot(df[mar])
```

In [26]:
```
#plot = sns.countplot(df[df["job"].isin(df["job"].value_counts().head(5).in
#plt.setp(plot.get_xticklabels(), rotation=90)
```

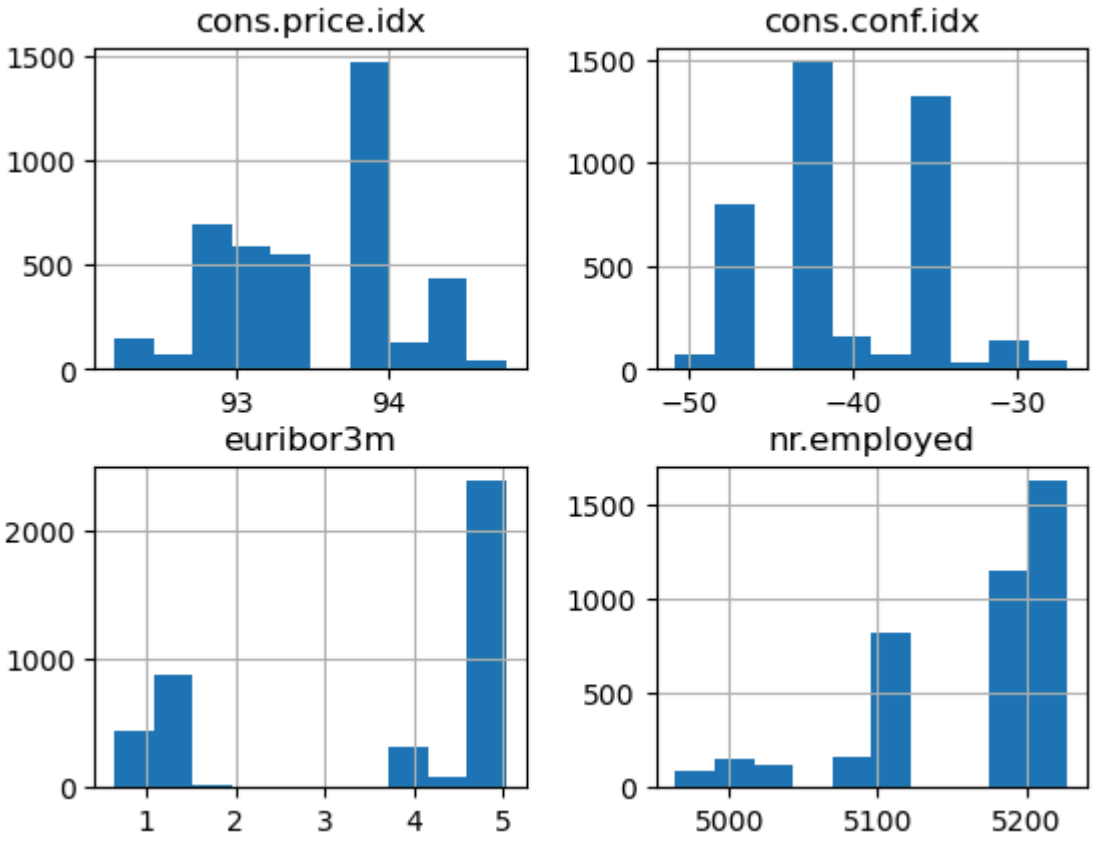
Visual Analysis of the feature interaction

Numerical Features

To analyze the interactionof numerical features , use hist ,pairplot, and heatmap plot functions:

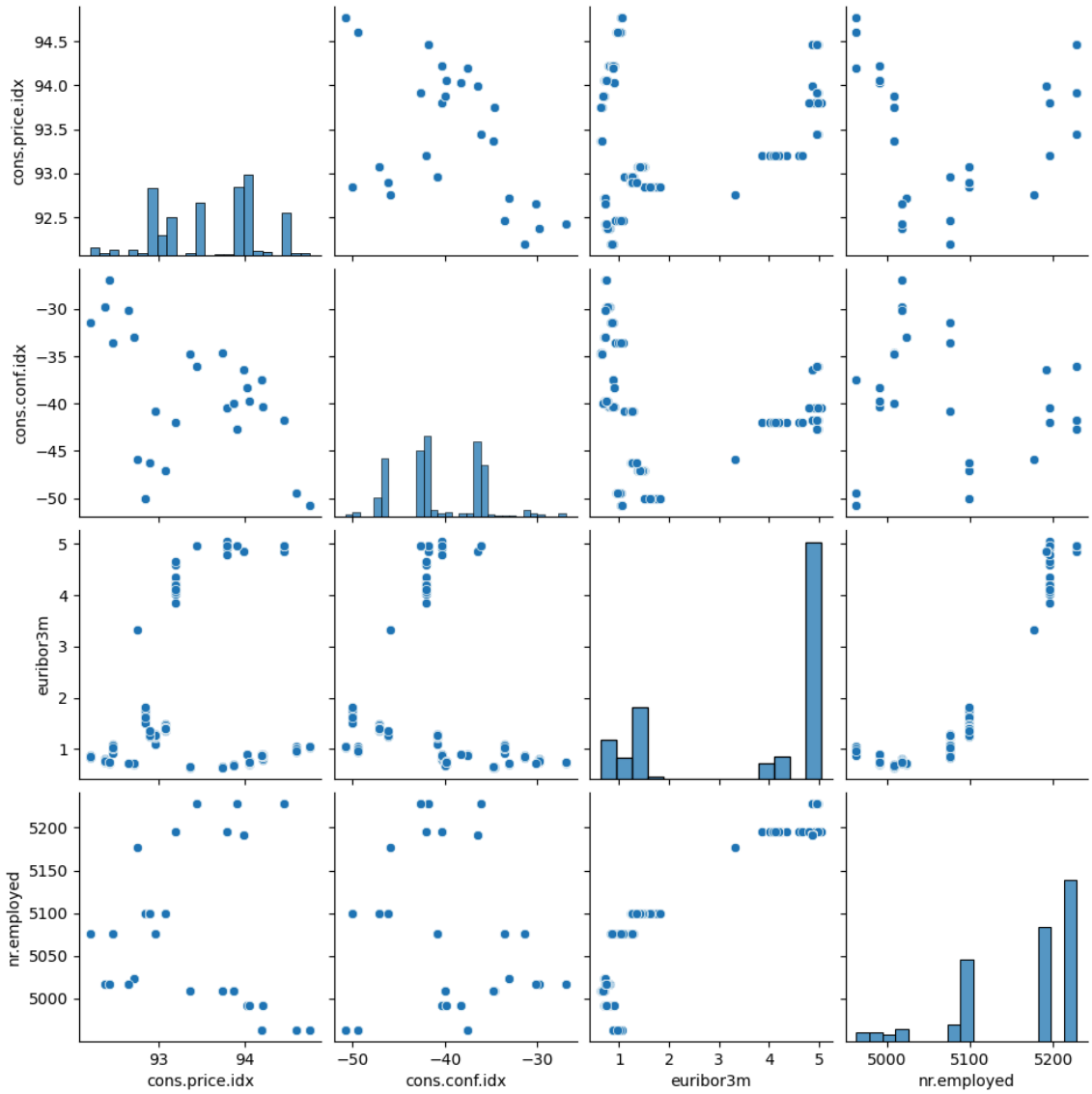
In [27]:
```
feat = ["cons.price.idx", "cons.conf.idx", "euribor3m", "nr.employed"]

df[feat].hist()
```

Out[27]:
```
array([[<Axes: title={'center': 'cons.price.idx'}>,
        <Axes: title={'center': 'cons.conf.idx'}>],
       [<Axes: title={'center': 'euribor3m'}>,
        <Axes: title={'center': 'nr.employed'}>]], dtype=object)
```



In [28]:
```
sns.pairplot(df[feat])
```

Out[28]:
```
<seaborn.axisgrid.PairGrid at 0x21459af5750>
```



In [29]:
```
sns.heatmap(df[feat].corr())
```

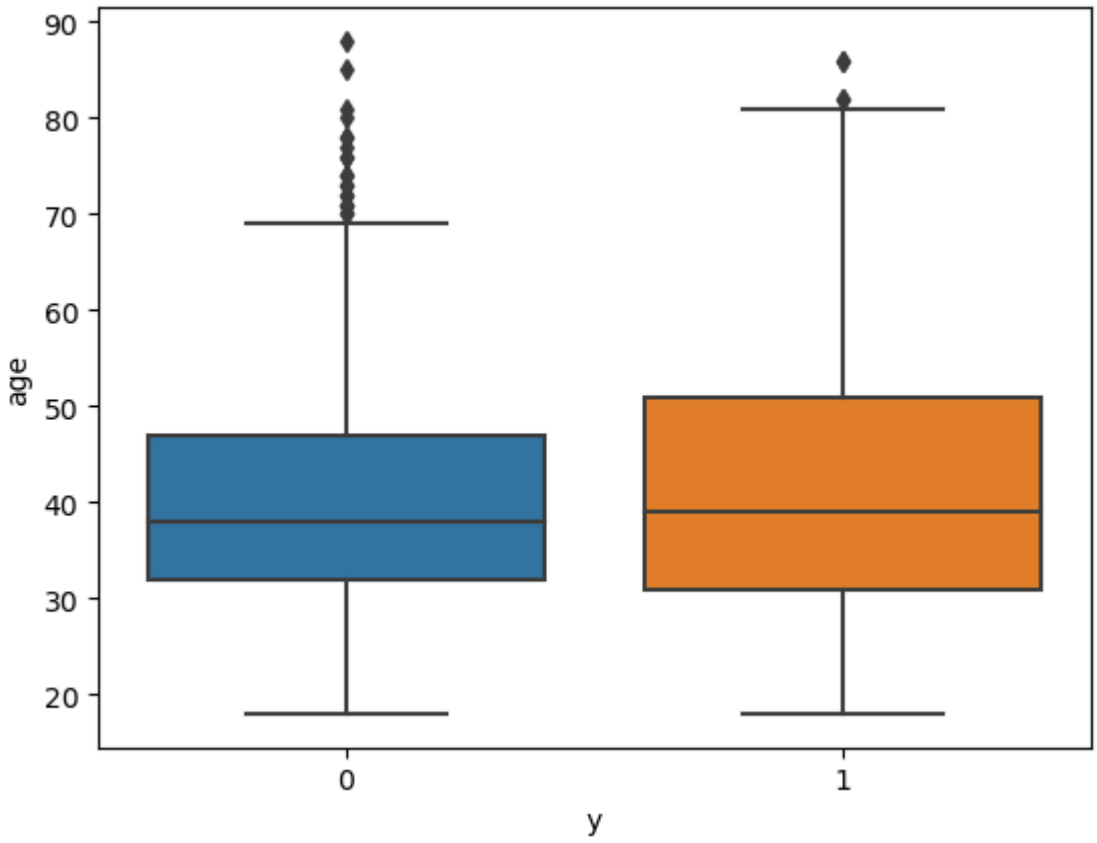
Out[29]: <Axes: >

Numerical And Categorical features

boxplot and violinplot are used for visual analysis of the numerical and categorical features

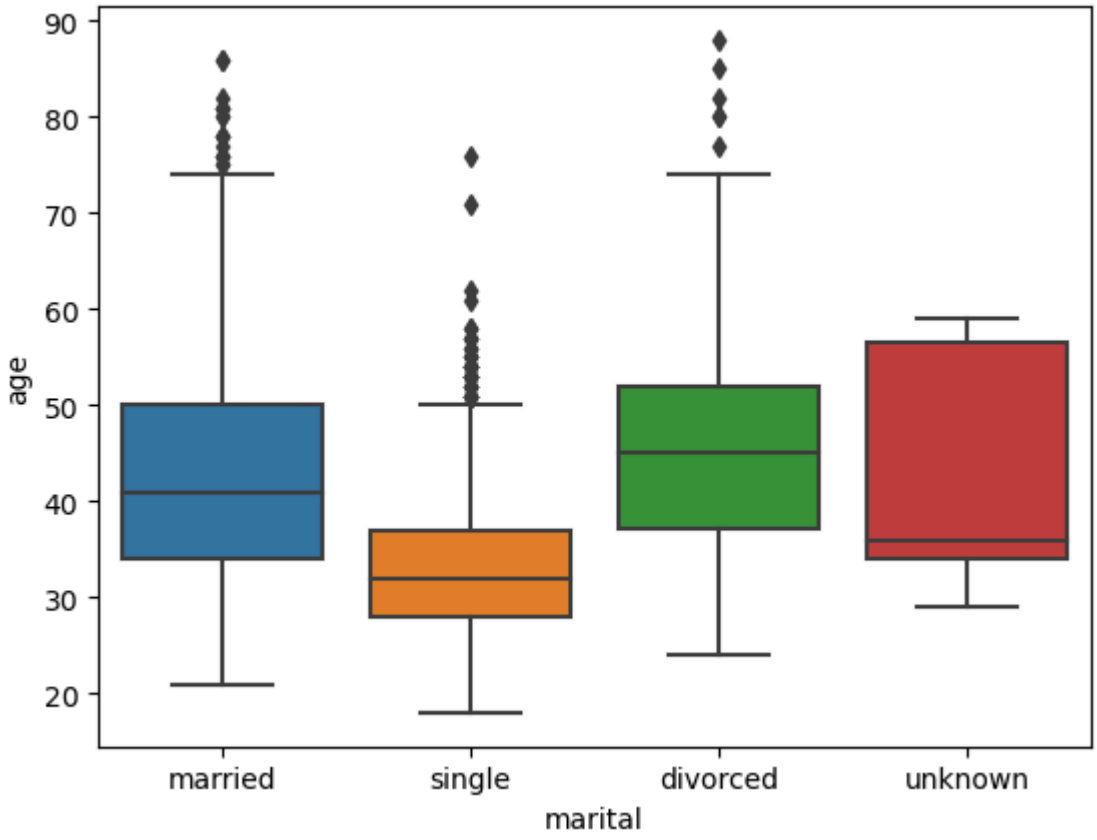
In [30]:
```
sns.boxplot(x='y', y='age', data =df)
```

Out[30]:
```
<Axes: xlabel='y', ylabel='age'>
```



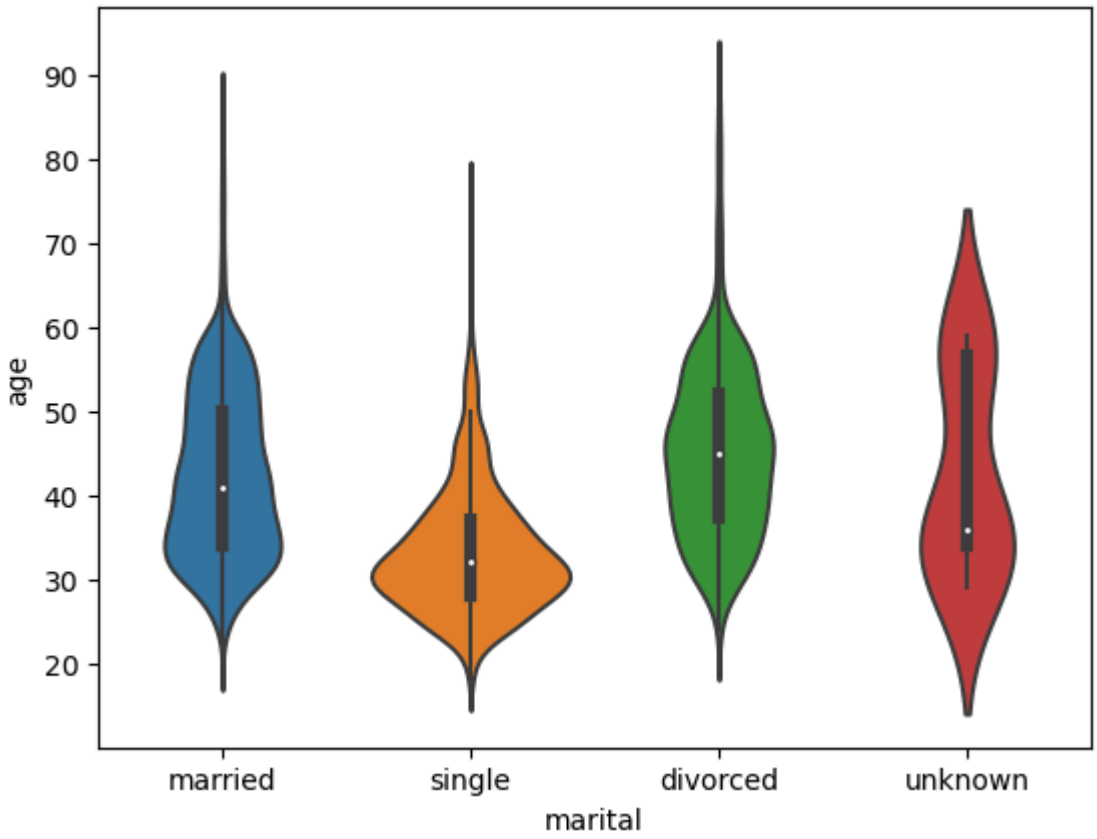
In [31]:
```
sns.boxplot(x='marital', y='age', data = df)
```

Out[31]: <Axes: xlabel='marital', ylabel='age'>



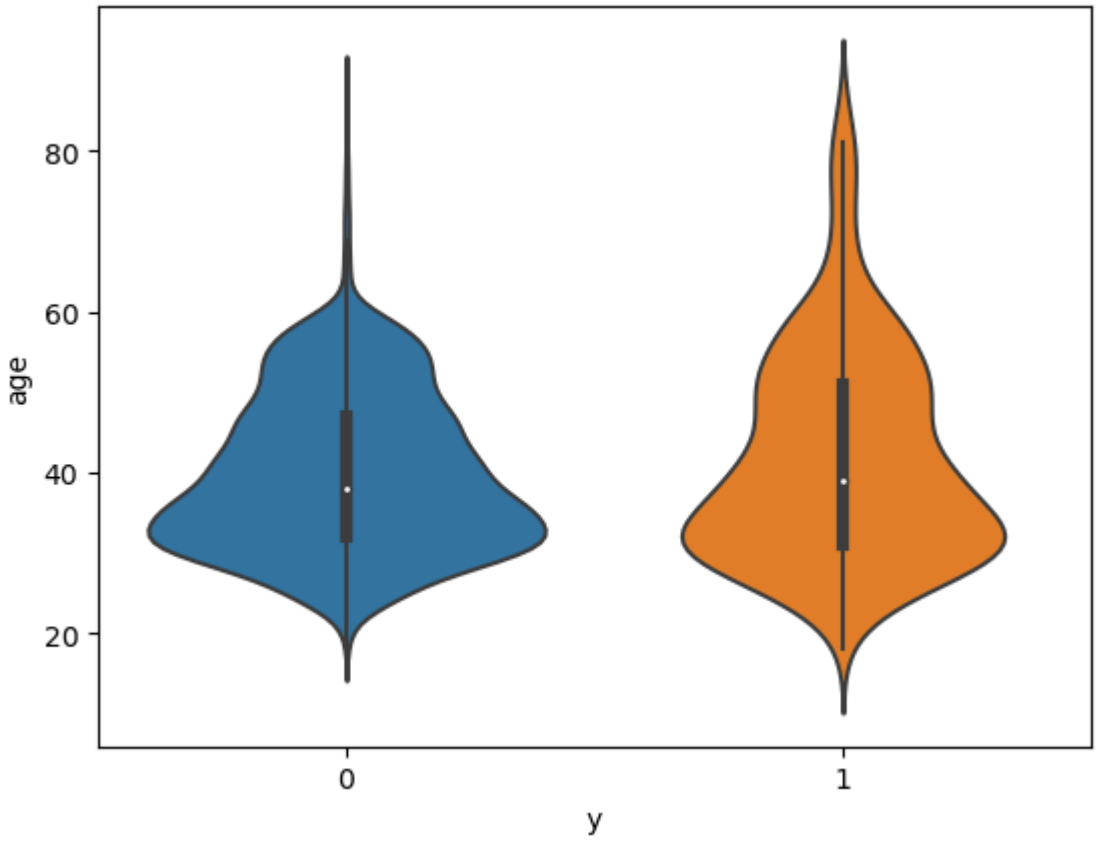
In [32]:
```
sns.violinplot(x='marital', y='age', data = df)
```

Out[32]: <Axes: xlabel='marital', ylabel='age'>



In [33]:
```
sns.violinplot(x='y', y='age', data = df)
```

Out[33]:
```
<Axes: xlabel='y', ylabel='age'>
```

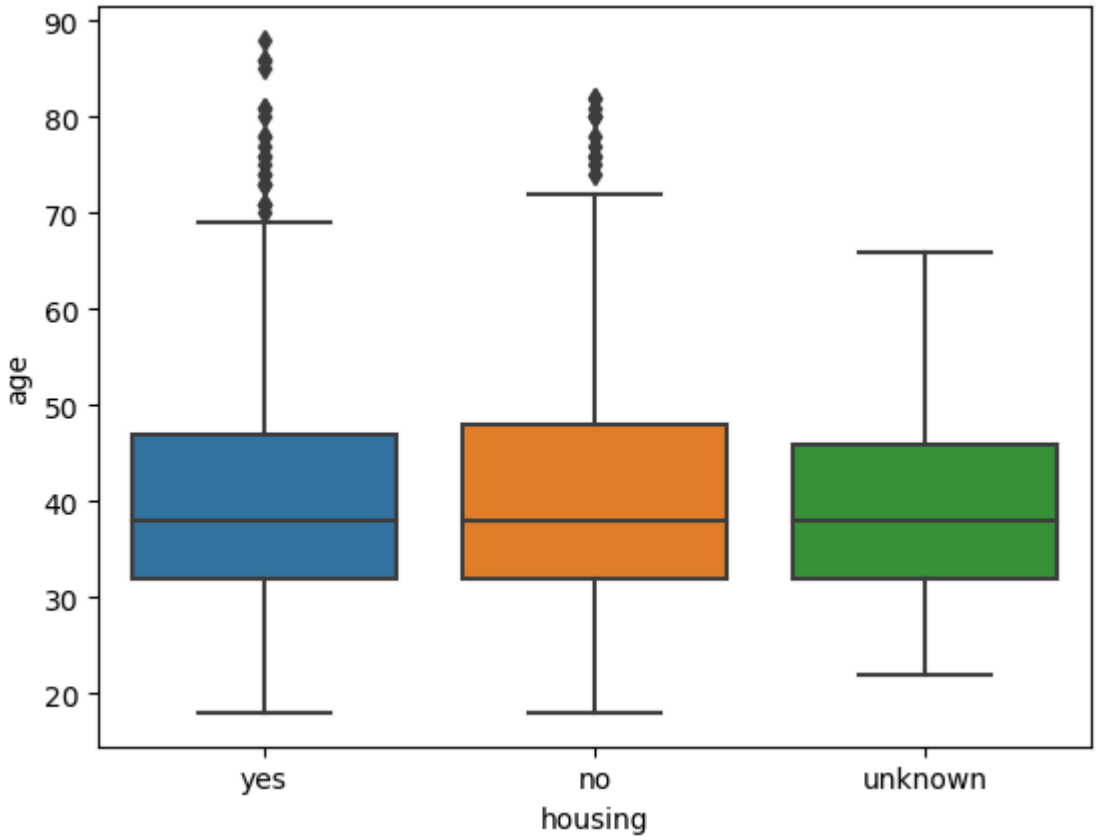


In [34]:
```
df.groupby("housing")["age"].mean()
```

Out[34]:
```
housing
no         40.213159
unknown    39.523810
yes        40.057931
Name: age, dtype: float64
```

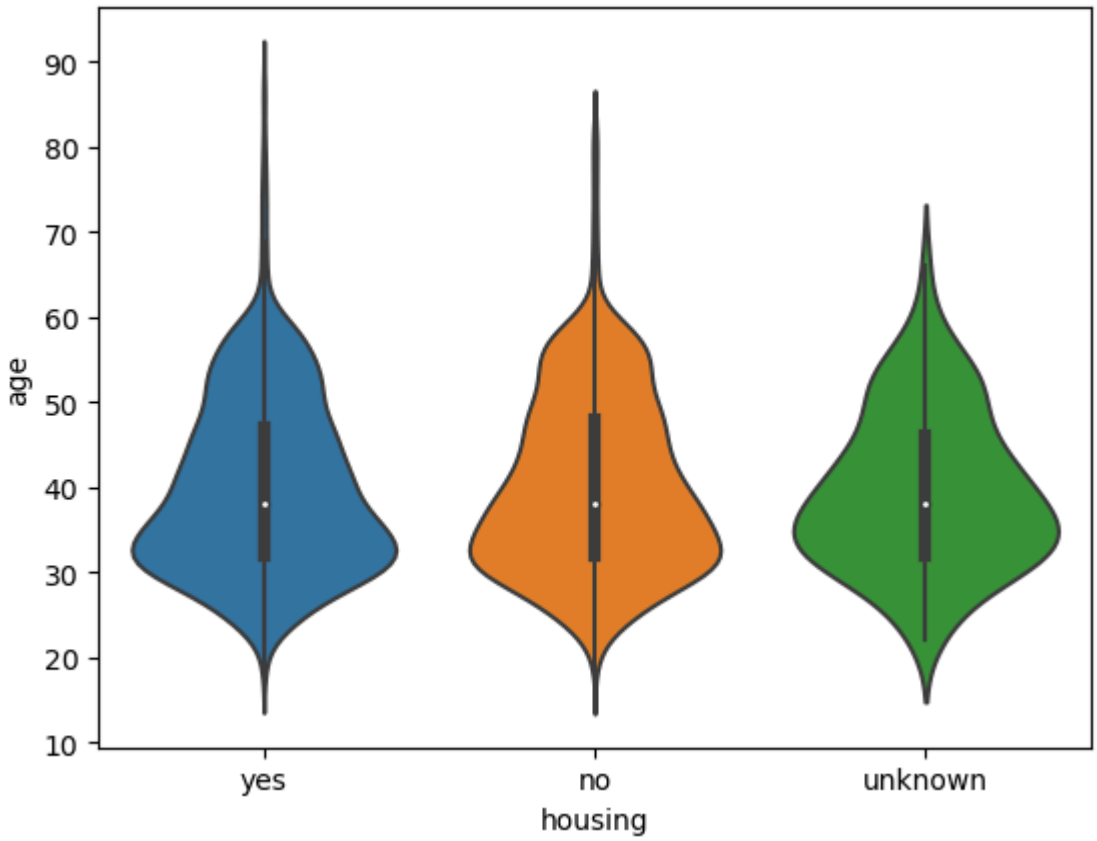
In [35]:
```
sns.boxplot(x='housing', y='age', data = df)
```

Out[35]:
```
<Axes: xlabel='housing', ylabel='age'>
```



In [36]:
```
sns.violinplot(x='housing', y='age', data = df)
```

Out[36]:
```
<Axes: xlabel='housing', ylabel='age'>
```



Categorical features

Use countplot for a visual interaction analysis between categorical features

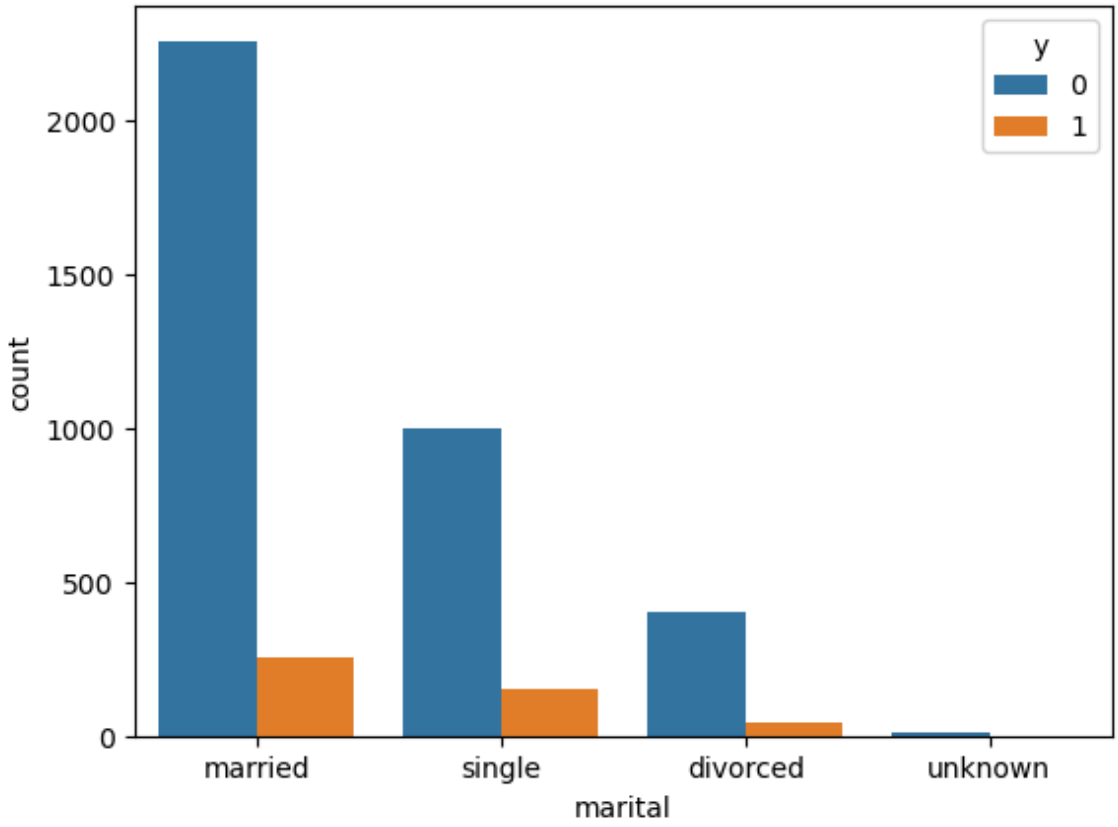
In [37]:
```
pd.crosstab(df["y"],df["marital"])
```

Out[37]:

marital	divorced	married	single	unknown
y				
0	403	2257	998	10
1	43	252	155	1

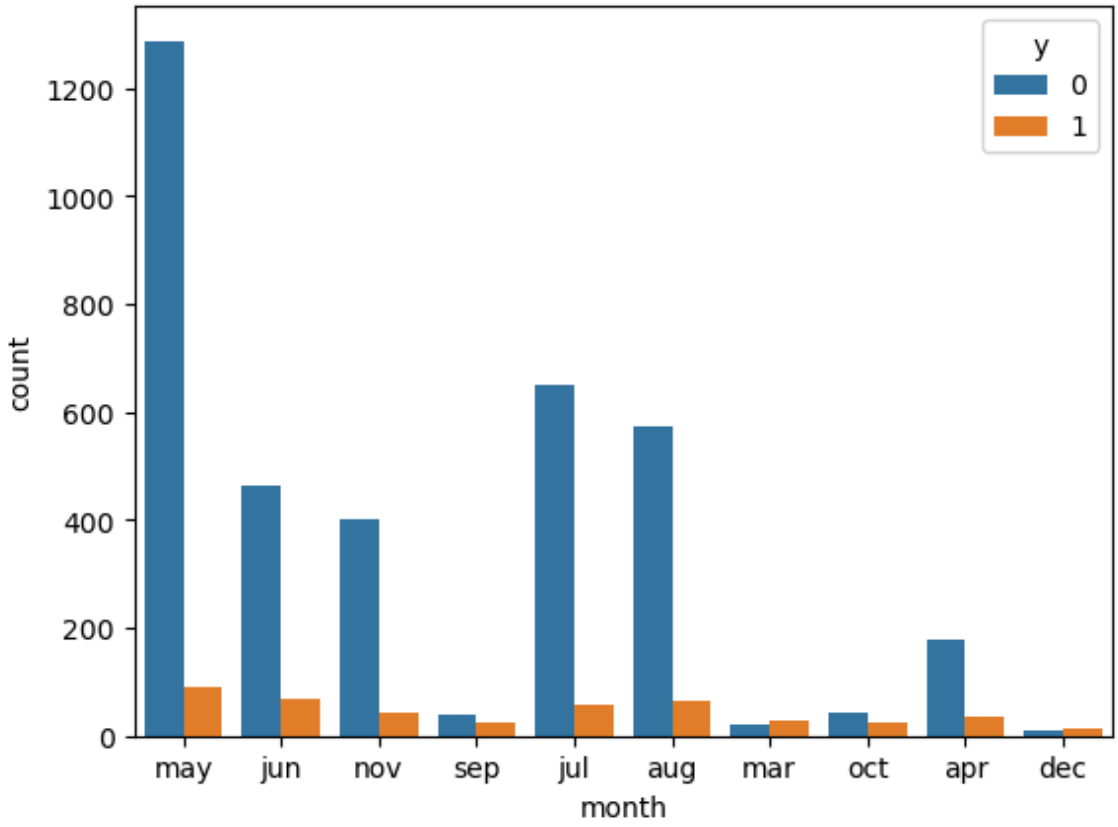
In [38]:
```
sns.countplot(x='marital', hue='y', data=df)
```

Out[38]:
```
<Axes: xlabel='marital', ylabel='count'>
```



In [39]:
```
sns.countplot(x="month", hue="y", data=df)
```

Out[39]:
```
<Axes: xlabel='month', ylabel='count'>
```

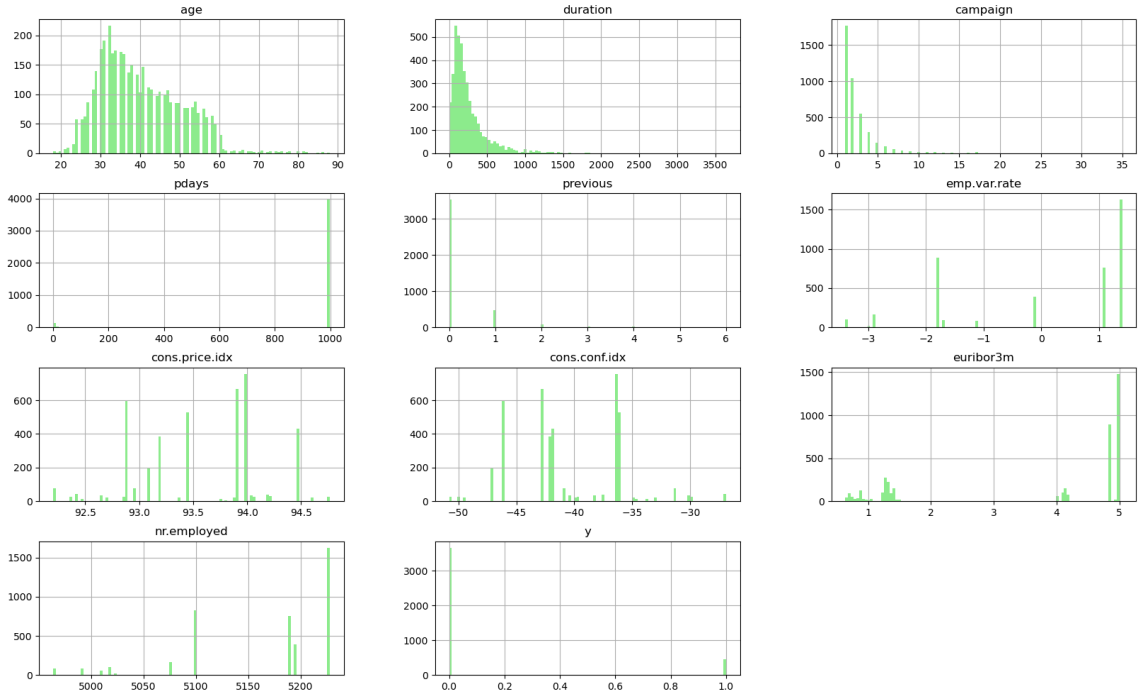


Comprehensive Visual analysis of the source banking dataset

Create the categorical and numerical lists for the correspondent dataset features

In [40]:
```
categorical = []
numerical = []
for feature in df.columns:
    if df[feature].dtype == object:
        categorical.append(feature)
    else:
        numerical.append(feature)

df[numerical].hist(figsize=(20,12), bins=100, color='lightgreen')
```

Out[40]:
```
array([[<Axes: title={'center': 'age'}>,
        <Axes: title={'center': 'duration'}>,
        <Axes: title={'center': 'campaign'}>],
       [<Axes: title={'center': 'pdays'}>,
        <Axes: title={'center': 'previous'}>,
        <Axes: title={'center': 'emp.var.rate'}>],
       [<Axes: title={'center': 'cons.price.idx'}>,
        <Axes: title={'center': 'cons.conf.idx'}>,
        <Axes: title={'center': 'euribor3m'}>],
       [<Axes: title={'center': 'nr.employed'}>,
        <Axes: title={'center': 'y'}>, <Axes: >]], dtype=object)
```

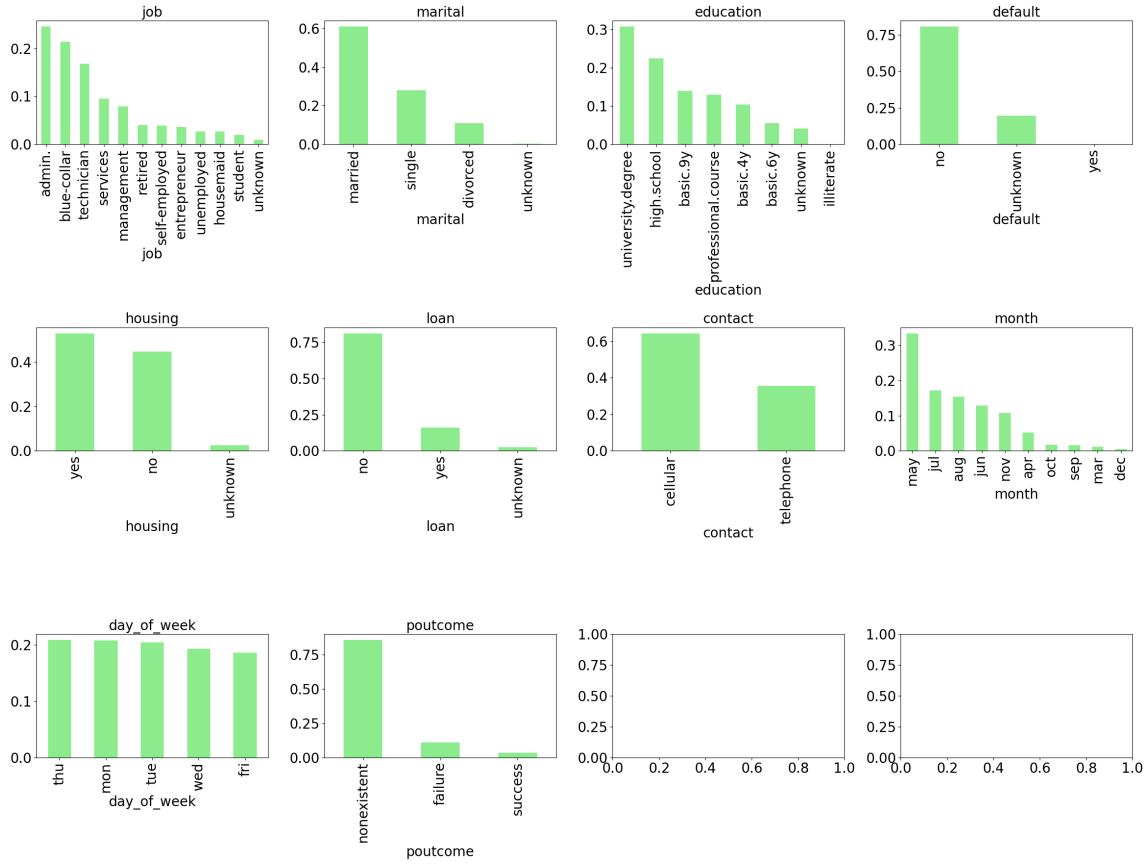


In [41]:
```
df.describe(include=['object'])
```

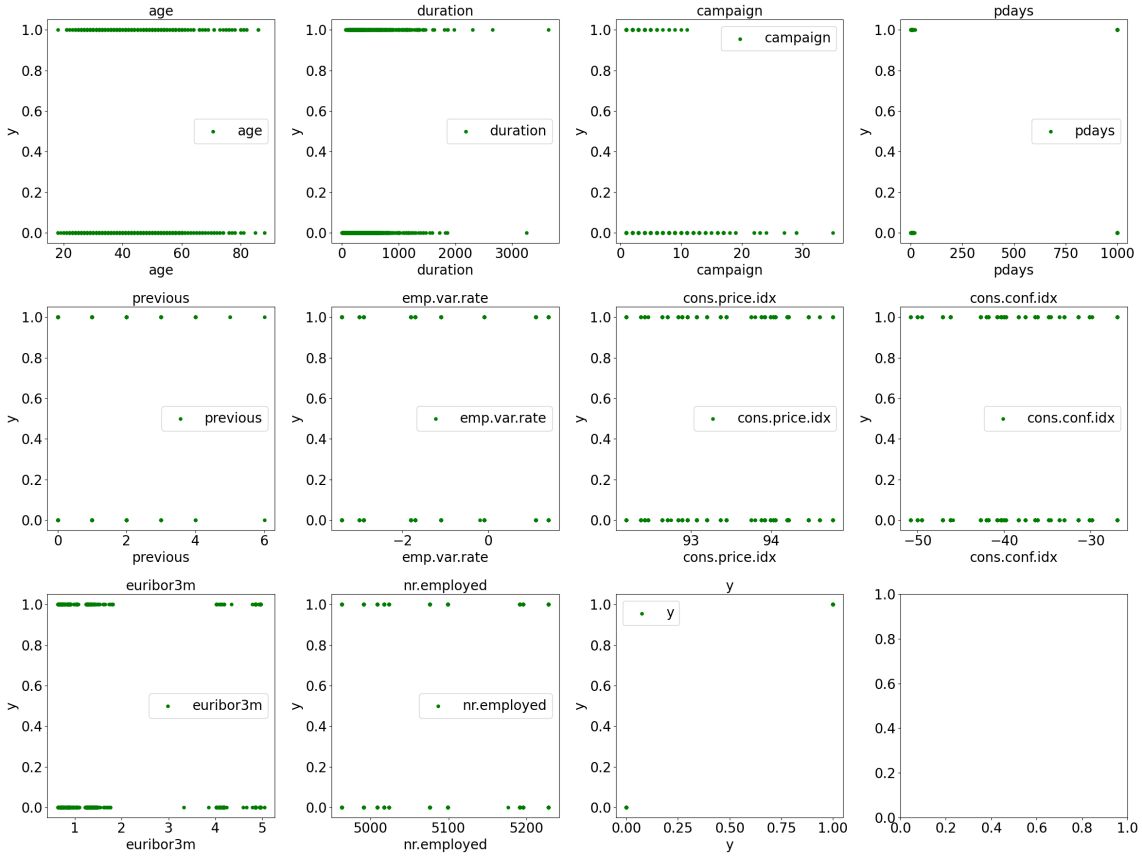
Out[41]:

	job	marital	education	default	housing	loan	contact	month	day_of_wee
count	4119	4119	4119	4119	4119	4119	4119	4119	411
unique	12	4	8	3	3	3	2	10	
top	admin.	married	university.degree	no	yes	no	cellular	may	th
freq	1012	2509	1264	3315	2175	3349	2652	1378	86

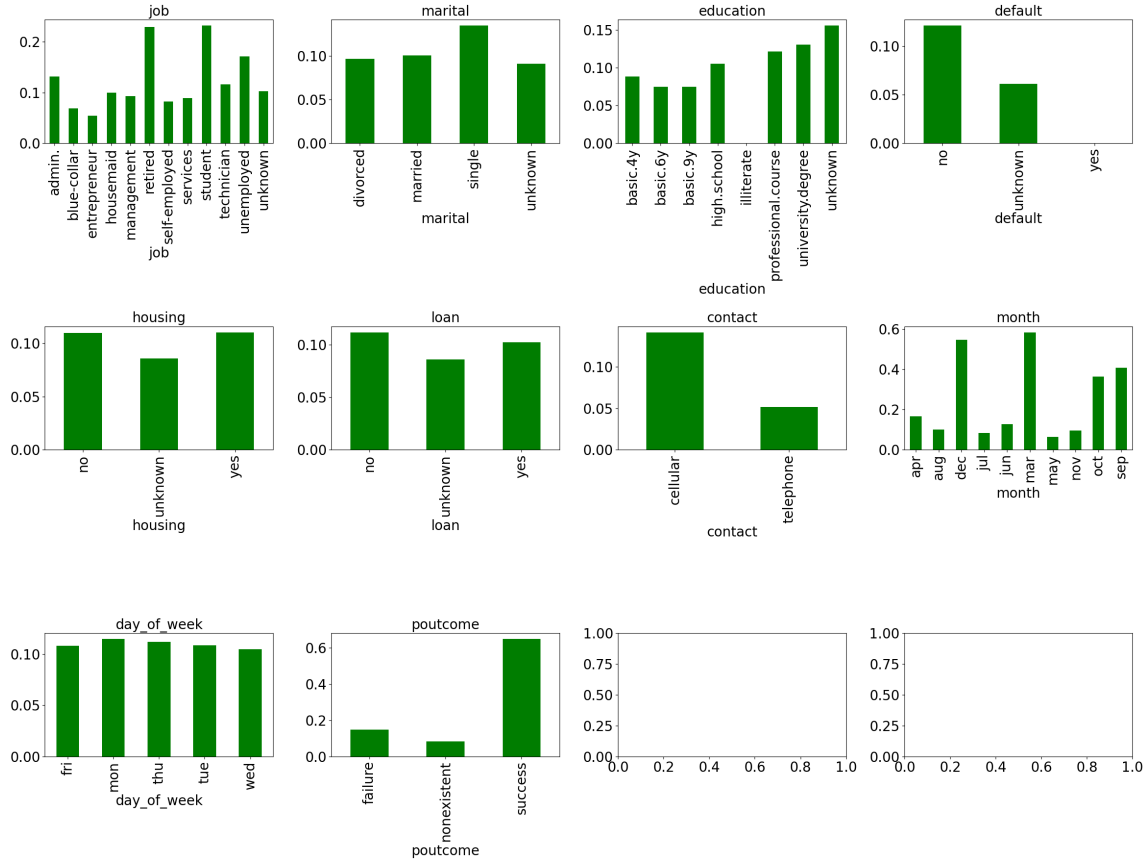
In [42]:
```
plt.rcParams['axes.labelsize'] = 20
plt.rcParams['axes.titlesize'] = 20
plt.rcParams['font.size'] = 20

fig, axes = plt.subplots(ncols=4, nrows = 3, figsize=(24, 18))
plt.subplots_adjust(left=None, bottom=None, right=None, top=None, wspace=No

for i in range(len(categorical)):
    df[categorical[i]].value_counts(normalize=True).plot(kind='bar', label=
    axes[i//4, i%4].set_title(categorical[i])
plt.tight_layout()
```



In [43]:
```
fig, axes = plt.subplots(ncols=4, nrows = 3, figsize=(24, 18))
plt.subplots_adjust(left=None, bottom=None, right=None, top=None, wspace=No

for i in range(len(numerical)):
    df.plot(x=numerical[i], y = 'y', label=numerical[i], ax=axes[i//4, i%4]
    axes[i//4, i%4].set_title(numerical[i])
plt.tight_layout()
```

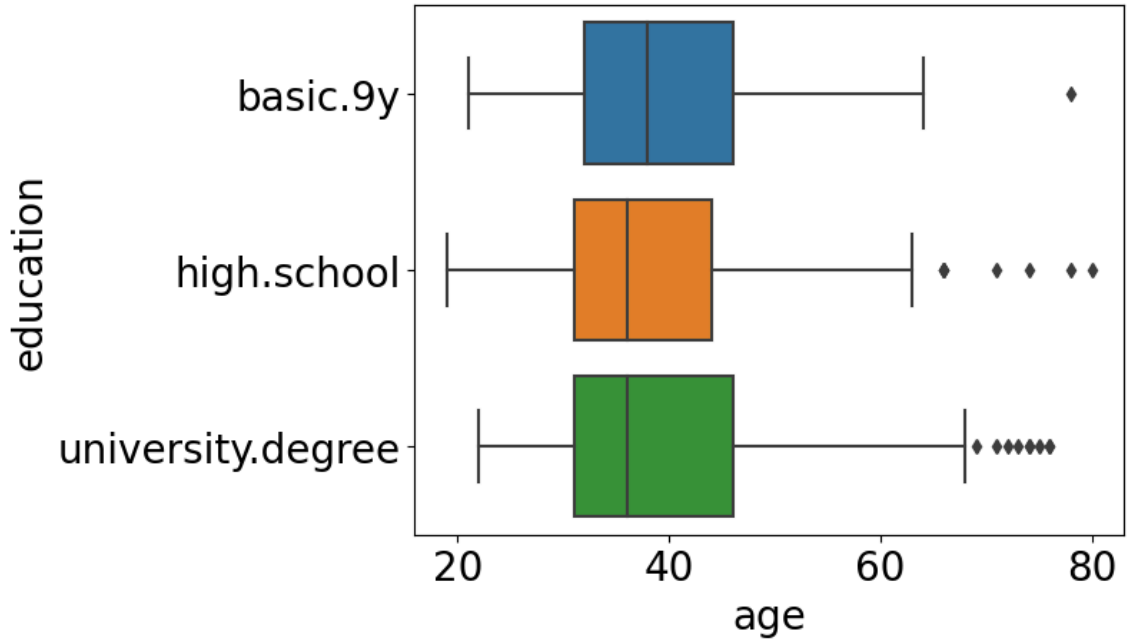


In [44]:
```
ig, axes = plt.subplots(ncols=4, nrows = 3, figsize=(24, 18))
plt.subplots_adjust(left=None, bottom=None, right=None, top=None, wspace=No

for i in range(len(categorical)):
    df.groupby(categorical[i])['y'].mean().plot(kind='bar', ax=axes[i//4, i
    axes[i//4, i%4].set_title(categorical[i])

plt.tight_layout()
```



In [45]:
```
top_3 = (
    df.education.value_counts().sort_values(ascending=False).head(3).index.
)
sns.boxplot(
    y="education", x="age", data=df[df.education.isin(top_3)], orient="h"
)
```

Out[45]: <Axes: xlabel='age', ylabel='education'>



In [46]:
```
df.default.value_counts()
```

Out[46]:
```
default
no         3315
unknown     803
yes           1
Name: count, dtype: int64
```

In [ ]:

In [ ]: