In [5]:
```
# Part-1
#import the useful libraries.
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [6]:
```
# Read the data set of "Marketing Analysis" in data.
marketing_data= pd.read_csv("marketing_analysis.csv")
```

In [7]:
```
# Printing the data
marketing_data
```

Out[7]:

	banking marketing	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed 6
0	customer id and age.	NaN	Customer salary and balance.	NaN	Customer marital status and job with education...	NaN	particula custome before targeted o no
1	customerid	age	salary	balance	marital	jobedu	targetec
2	1	58	100000	2143	married	management,tertiary	yes
3	2	44	60000	29	single	technician,secondary	yes
4	3	33	120000	2	married	entrepreneur,secondary	yes
...	...	...	...	...	...	...	..
45208	45207	51.0	60000	825	married	technician,tertiary	yes
45209	45208	71.0	55000	1729	divorced	retired,primary	yes
45210	45209	72.0	55000	5715	married	retired,secondary	yes
45211	45210	57.0	20000	668	married	blue-collar,secondary	yes
45212	45211	37.0	120000	2971	married	entrepreneur,secondary	yes

45213 rows × 19 columns

In [8]:
```
# Read the file in data without first two rows as it is of no use.
marketing_data = pd.read_csv("marketing_analysis.csv",skiprows = 2)
#print the head of the data frame.
marketing_data.head()
```

Out[8]:

	customerid	age	salary	balance	marital	jobedu	targeted	default	housing
0	1	58.0	100000	2143	married	management,tertiary	yes	no	yes

	customerid	age	salary	balance	marital	jobedu	targeted	default	housing
1	2	44.0	60000	29	single	technician,secondary	yes	no	yes
2	3	33.0	120000	2	married	entrepreneur,secondary	yes	no	yes
3	4	47.0	20000	1506	married	blue-collar,unknown	no	no	yes

In [9]:
```
#Data Cleaning
# Drop the customer id as it is of no use.
marketing_data.drop('customerid', axis = 1, inplace = True)
```

In [10]:
```
marketing_data
```

Out[10]:

	age	salary	balance	marital	jobedu	targeted	default	housing	loan
0	58.0	100000	2143	married	management,tertiary	yes	no	yes	no
1	44.0	60000	29	single	technician,secondary	yes	no	yes	no
2	33.0	120000	2	married	entrepreneur,secondary	yes	no	yes	yes
3	47.0	20000	1506	married	blue-collar,unknown	no	no	yes	no
4	33.0	0	1	single	unknown,unknown	no	no	no	no
...	...	...	...	...	...	...	...	...	...
45206	51.0	60000	825	married	technician,tertiary	yes	no	no	no
45207	71.0	55000	1729	divorced	retired,primary	yes	no	no	no
45208	72.0	55000	5715	married	retired,secondary	yes	no	no	no
45209	57.0	20000	668	married	blue-collar,secondary	yes	no	no	no t
45210	37.0	120000	2971	married	entrepreneur,secondary	yes	no	no	no

45211 rows × 18 columns

In [11]:
```
#Extract job & Education in newly from "jobedu" column.
marketing_data['job']= marketing_data["jobedu"].apply(lambda x:
x.split(",")[0])
marketing_data['education']= marketing_data["jobedu"].apply(lambda x:
x.split(",")[1])
# Drop the "jobedu" column from the dataframe.
marketing_data.drop('jobedu', axis = 1, inplace = True)
# Printing the Dataset
marketing_data
```

Out[11]:

	age	salary	balance	marital	targeted	default	housing	loan	contact	day	month
0	58.0	100000	2143	married	yes	no	yes	no	unknown	5	may, 2017
1	44.0	60000	29	single	yes	no	yes	no	unknown	5	may, 2017
2	33.0	120000	2	married	yes	no	yes	yes	unknown	5	may, 2017
3	47.0	20000	1506	married	no	no	yes	no	unknown	5	may, 2017
4	33.0	0	1	single	no	no	no	no	unknown	5	may, 2017
...	...	...	...	...	...	...	...	...	...	...	...
45206	51.0	60000	825	married	yes	no	no	no	cellular	17	nov, 2017
45207	71.0	55000	1729	divorced	yes	no	no	no	cellular	17	nov, 2017
45208	72.0	55000	5715	married	yes	no	no	no	cellular	17	nov, 2017
45209	57.0	20000	668	married	yes	no	no	no	telephone	17	nov, 2017
45210	37.0	120000	2971	married	yes	no	no	no	cellular	17	nov, 2017

45211 rows × 19 columns

In [12]:
```
marketing_data['job']
```

Out[12]:
```
0          management
1          technician
2        entrepreneur
3         blue-collar
4             unknown
              ...
45206      technician
45207         retired
45208         retired
45209     blue-collar
45210    entrepreneur
Name: job, Length: 45211, dtype: object
```

In [13]:
```
marketing_data['education']
```

Out[13]:
```
0          tertiary
1         secondary
2         secondary
3           unknown
4           unknown
             ...
45206      tertiary
45207       primary
45208     secondary
45209     secondary
45210     secondary
Name: education, Length: 45211, dtype: object
```

In [14]:
```
# Checking the missing values
marketing_data.isnull().sum()
```

Out[14]:
```
age           20
salary         0
balance        0
marital        0
targeted       0
default        0
housing        0
loan           0
contact        0
day            0
month         50
duration       0
campaign       0
pdays          0
previous       0
poutcome       0
response      30
job            0
education      0
dtype: int64
```

In [15]:
```
marketing_data.isnull()
```

Out[15]:

	age	salary	balance	marital	targeted	default	housing	loan	contact	day	month
0	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...
45206	False	False	False	False	False	False	False	False	False	False	False
45207	False	False	False	False	False	False	False	False	False	False	False

	age	salary	balance	marital	targeted	default	housing	loan	contact	day	month
45208	False	False	False	False	False	False	False	False	False	False	False
45209	False	False	False	False	False	False	False	False	False	False	False
45210	False	False	False	False	False	False	False	False	False	False	False

In [16]:
```
#age value with null()
marketing_data[marketing_data.age.isnull()]
```

Out[16]:

	age	salary	balance	marital	targeted	default	housing	loan	contact	day	month
161	NaN	60000	47	single	no	no	yes	no	unknown	5	may, 2017
1021	NaN	60000	-500	married	yes	no	yes	no	unknown	7	may, 2017
1585	NaN	100000	123	divorced	yes	no	no	no	unknown	8	may, 2017
3943	NaN	70000	62	married	yes	no	yes	yes	unknown	16	may, 2017
5675	NaN	20000	-85	married	yes	no	yes	yes	unknown	26	may, 2017
6873	NaN	70000	1336	divorced	yes	no	yes	no	unknown	28	may, 2017
9692	NaN	50000	-162	single	no	no	yes	yes	unknown	6	jun, 2017
11653	NaN	8000	2562	single	yes	no	no	no	unknown	20	jun, 2017
13417	NaN	20000	452	divorced	yes	no	yes	no	cellular	9	jul, 2017
14541	NaN	20000	-84	married	yes	no	no	yes	cellular	15	jul, 2017
17159	NaN	120000	-2082	married	yes	yes	no	yes	cellular	28	jul, 2017
19457	NaN	60000	1055	married	yes	no	yes	no	cellular	7	aug, 2017
23483	NaN	120000	456	married	yes	no	yes	yes	cellular	28	aug, 2017
23939	NaN	20000	85	married	yes	no	no	no	cellular	29	aug, 2017
26531	NaN	60000	8112	divorced	yes	no	yes	yes	cellular	20	nov, 2017
35920	NaN	20000	1	divorced	yes	no	yes	no	cellular	8	may, 2017
40757	NaN	55000	3444	divorced	no	no	no	no	cellular	10	aug, 2017
40994	NaN	100000	3371	single	no	no	yes	yes	cellular	13	aug, 2017

	age	salary	balance	marital	targeted	default	housing	loan	contact	day	month
43573	NaN	16000	962	married	yes	no	yes	no	cellular	28	apr, 2017

In [17]:
```
#age value with not null(~)
marketing_data=marketing_data[~marketing_data.age.isnull()]
```

In [18]:
```
#deleting null values from months columns

marketing_data=marketing_data[~marketing_data.month.isnull()]
```

In [19]:
```
marketing_data.head()
```

Out[19]:

	age	salary	balance	marital	targeted	default	housing	loan	contact	day	month	durati
0	58.0	100000	2143	married	yes	no	yes	no	unknown	5	may, 2017	261 s
1	44.0	60000	29	single	yes	no	yes	no	unknown	5	may, 2017	151 s
2	33.0	120000	2	married	yes	no	yes	yes	unknown	5	may, 2017	76 s
3	47.0	20000	1506	married	no	no	yes	no	unknown	5	may, 2017	92 s
4	33.0	0	1	single	no	no	no	no	unknown	5	may, 2017	198 s

In [20]:
```
marketing_data.isnull().sum()
```

Out[20]:
```
age           0
salary        0
balance       0
marital       0
targeted      0
default       0
housing       0
loan          0
contact       0
day           0
month         0
duration      0
campaign      0
pdays         0
previous      0
poutcome      0
response     30
job           0
education     0
dtype: int64
```

In [21]:
```
# Find the mode of month in data
month_mode = marketing_data.month.mode()[0]
```

In [22]:
```
# Fill the missing values with mode value of month in data.
marketing_data.month.fillna(month_mode, inplace = True)
```

In [23]:
```
# Let's see the null values in the month column.
marketing_data.month.isnull().sum()
```

Out[23]:
```
0
```

In [24]:
```
marketing_data.marital.unique()
```

Out[24]:
```
array(['married', 'single', 'divorced'], dtype=object)
```

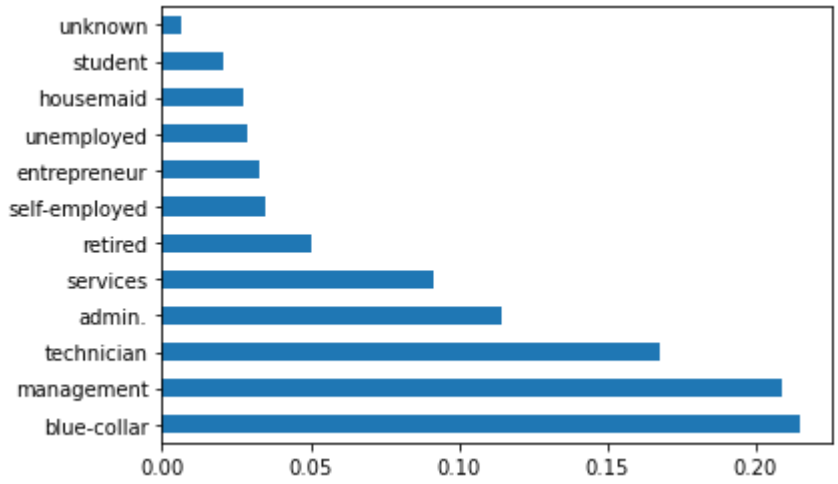
In [25]:
```
# Normalize-Let's calculate the percentage of each job status category.
marketing_data.job.value_counts(normalize=True)
```

Out[25]:
```
blue-collar      0.215281
management       0.209255
technician       0.168051
admin.           0.114397
services         0.091846
retired          0.050043
self-employed    0.034913
entrepreneur     0.032875
unemployed       0.028843
housemaid        0.027359
student          0.020757
unknown          0.006380
Name: job, dtype: float64
```

In [26]:
```
# Let's calculate the percentage of each job status category.
marketing_data.job.value_counts()
```

Out[26]:
```
blue-collar      9718
management       9446
technician       7586
admin.           5164
services         4146
retired          2259
self-employed    1576
entrepreneur     1484
unemployed       1302
housemaid        1235
student           937
unknown           288
Name: job, dtype: int64
```

In [27]:
```
#plot the bar graph of percentage job categories
marketing_data.job.value_counts(normalize=True).plot.barh()
plt.show()
```

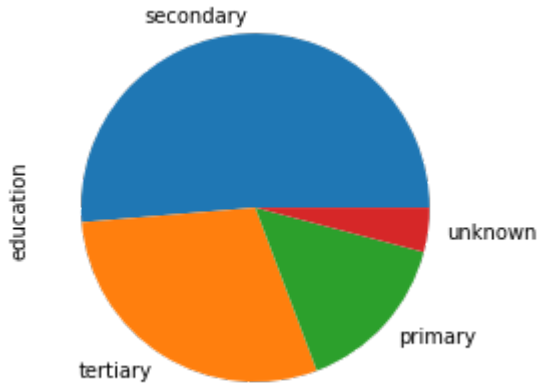


In [28]:
```
# Dropping the records with age missing in data dataframe.
marketing_data = marketing_data[~marketing_data.age.isnull()].copy()
```

In [29]:
```
# Checking the missing values in the dataset.
marketing_data.isnull().sum()
```

Out[29]:
```
age           0
salary        0
balance       0
marital       0
targeted      0
default       0
housing       0
loan          0
contact       0
day           0
month         0
duration      0
campaign      0
pdays         0
previous      0
poutcome      0
response     30
job           0
education     0
dtype: int64
```

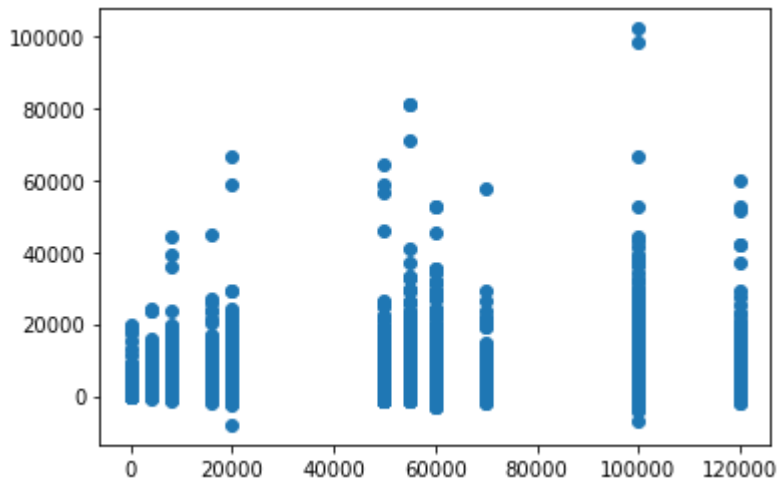
In [32]:
```
#plot the pie graph of percentage job categories
marketing_data.education.value_counts(normalize=True).plot.pie()
plt.show()
```



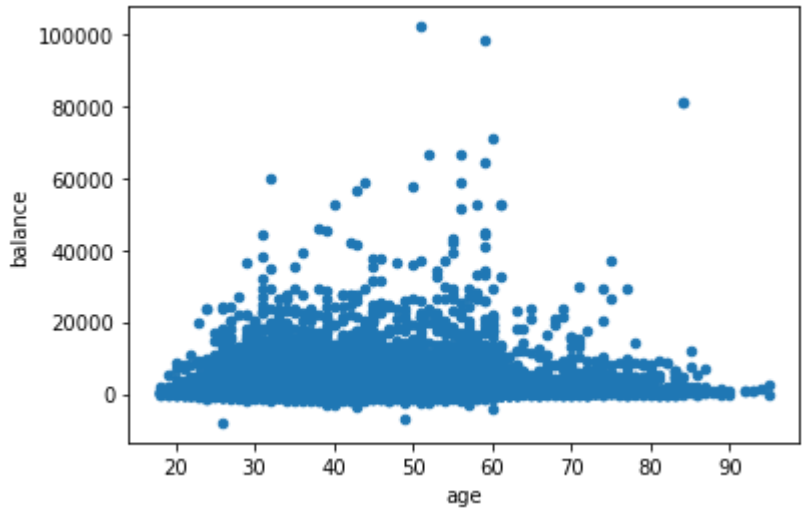
In [33]:
```
marketing_data.salary.describe()
```

Out[33]:
```
count     45141.000000
mean      57006.911677
std       32086.873882
min           0.000000
25%       20000.000000
50%       60000.000000
75%       70000.000000
max      120000.000000
Name: salary, dtype: float64
```

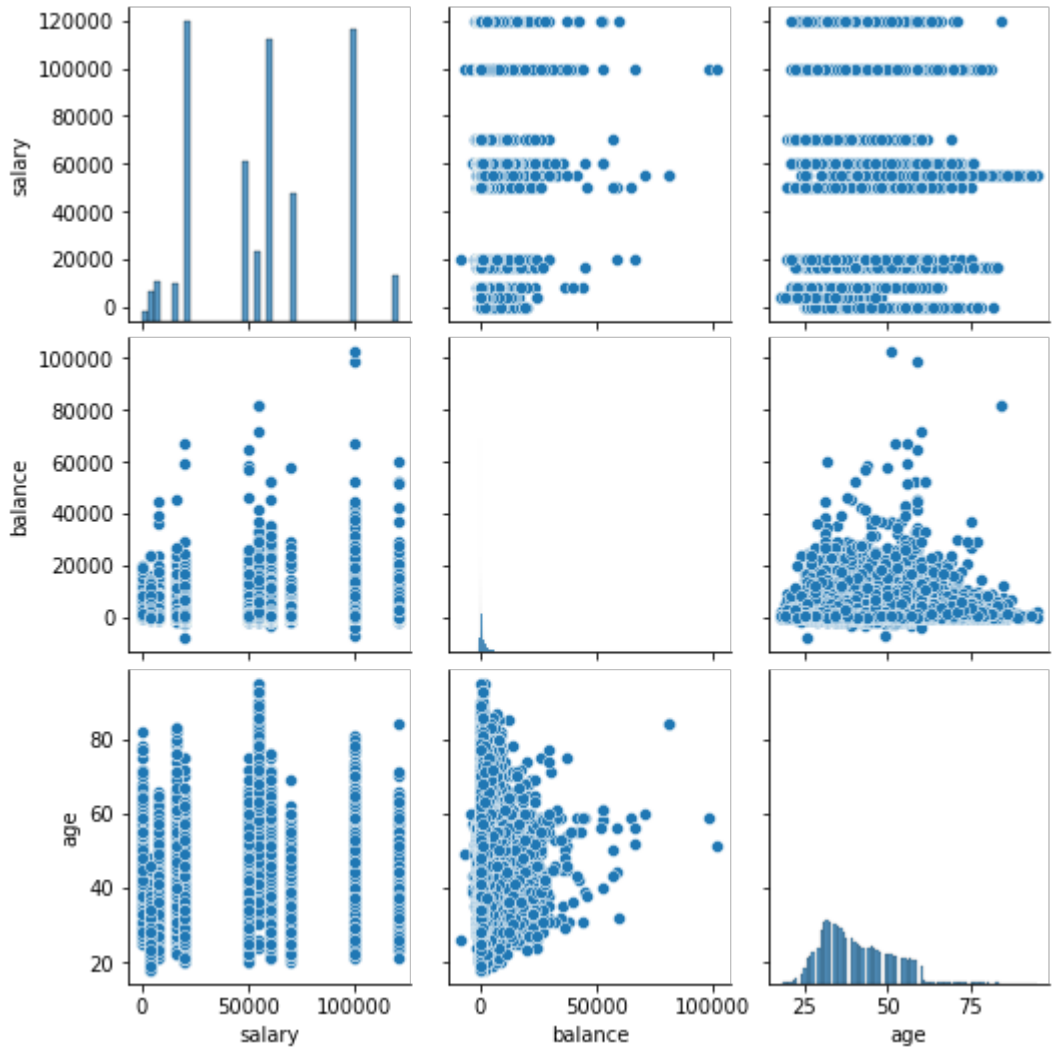
In [34]:
```
plt.scatter(marketing_data.salary, marketing_data.balance)
plt.show()
```



In [35]:
```
marketing_data.plot.scatter(x='age',y='balance')
plt.show()
```



In [37]:
```
sns.pairplot(data=marketing_data, vars=['salary','balance','age'])
plt.show()
```



In [38]:
```
marketing_data[['salary','balance','age']].corr()
```

Out[38]:

	salary	balance	age
salary	1.000000	0.055223	0.024419
balance	0.055223	1.000000	0.097847
age	0.024419	0.097847	1.000000

In [39]:
```
sns.heatmap(marketing_data[['salary','balance','age']].corr(), annot=True
plt.show()
```

In [40]:
```
marketing_data.groupby('response')['salary'].mean()
```

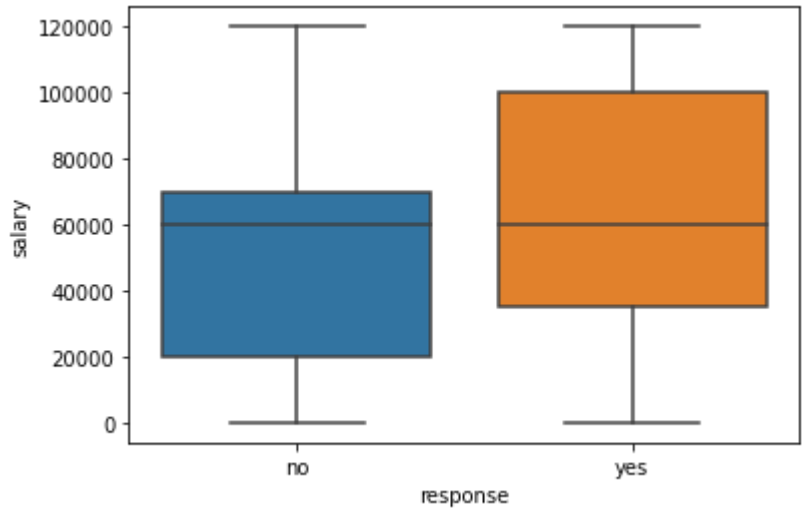
Out[40]:
```
response
no     56771.917052
yes    58770.411063
Name: salary, dtype: float64
```

In [41]:
```
marketing_data.groupby('response')['salary'].median()
```

Out[41]:
```
response
no     60000.0
yes    60000.0
Name: salary, dtype: float64
```

In [42]:
```
sns.boxplot(x='response', y='salary', data=marketing_data)
```

Out[42]: <AxesSubplot:xlabel='response', ylabel='salary'>



In [43]:
```
marketing_data['response_rate']=np.where(marketing_data.response=='yes',1
```

In [44]:
```
marketing_data['response_rate']
```

Out[44]:
```
0         0
1         0
2         0
3         0
4         0
        ..
45206     1
45207     1
45208     1
45209     0
45210     0
Name: response_rate, Length: 45141, dtype: int32
```

In [45]:
```
marketing_data['response']
```

Out[45]:
```
0         no
1         no
2         no
3         no
4         no
        ...
45206    yes
45207    yes
45208    yes
45209     no
45210     no
Name: response, Length: 45141, dtype: object
```

In [46]:
```
marketing_data.response_rate.value_counts()
```

Out[46]:

```
0    39862
1     5279
```

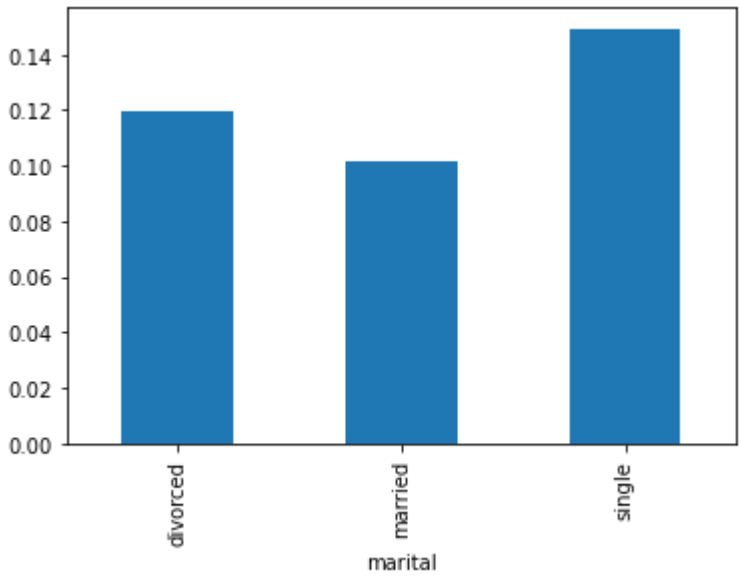
In [47]:
```
marketing_data.groupby('marital')['response_rate'].mean()
```

Out[47]:
```
marital
divorced    0.119277
married     0.101240
single      0.149413
Name: response_rate, dtype: float64
```

In [48]:
```
#marketing_data.groupby('marital')['response'].mean()
```

In [49]:
```
marketing_data.groupby('marital')['response_rate'].mean().plot.bar()
```

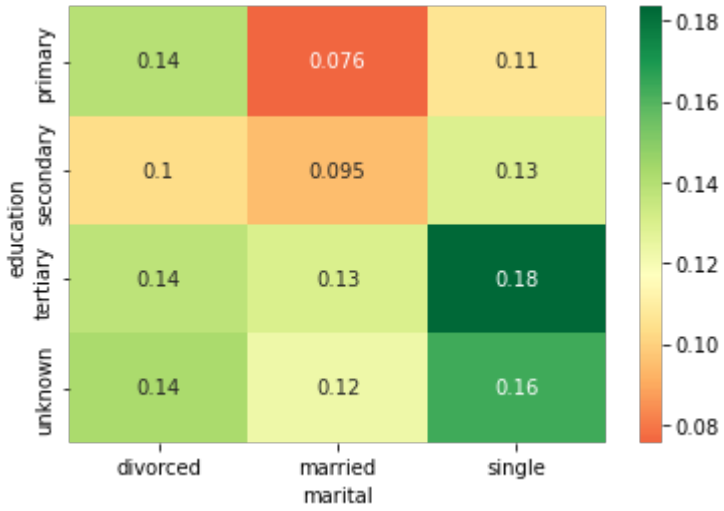
Out[49]:
```
<AxesSubplot:xlabel='marital'>
```



In [50]:
```
#create pivot table
result = pd.pivot_table(data=marketing_data, index='education',
columns='marital',values='response_rate')
print(result)
```

```
marital    divorced   married    single
education
primary    0.138852  0.075616  0.105758
secondary  0.103166  0.094663  0.129349
tertiary   0.137509  0.129661  0.183396
unknown    0.142012  0.122837  0.163188
```

In [51]:
```
#create heat map of education vs marital vs response_rate
sns.heatmap(result, annot=True, cmap = 'RdYlGn', center=0.117)
plt.show()
```



In [1]:
```
#Part-2
#Again loading and reshaping, checking the dataset for Answering the questions
#import the useful libraries.
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import scipy
from scipy import stats
marketing_data = pd.read_csv("marketing_analysis.csv",skiprows = 2)
marketing_data['job']= marketing_data["jobedu"].apply(lambda x:
x.split(",")[0])
marketing_data['education']= marketing_data["jobedu"].apply(lambda x:
x.split(",")[1])
# Drop the "jobedu" column from the dataframe.
marketing_data.drop('jobedu', axis = 1, inplace = True)
# Printing the Dataset
#marketing_data
marketing_data.isnull().sum()
```

Out[1]:
```
customerid     0
age           20
salary         0
balance        0
marital        0
targeted       0
default        0
housing        0
loan           0
contact        0
day            0
month         50
duration       0
campaign       0
pdays          0
previous       0
poutcome       0
response      30
job            0
education      0
dtype: int64
```

In [2]:
```
#Question 1: Implement three alternative approach(es) to handle missing values
#age and/or month columns. Explain why you decided to implement these approach
marketing_data['age'] = marketing_data.age.fillna(marketing_data.age.mean())
print('\nIn this way missing values of age column are filled-up with the avera
# Checking the missing values
marketing_data.isnull().sum()
```

```
In this way missing values of age column are filled-up with the average, va
lue of ages
so that an approaximate analyzing can be performed.
```

Out[2]:

```
customerid     0
age            0
salary         0
balance        0
marital        0
targeted       0
default        0
housing        0
loan           0
contact        0
day            0
month         50
duration       0
campaign       0
pdays          0
previous       0
poutcome       0
response      30
job            0
education      0
```

In [3]:
```
#Again re-loading, checking the dataset for answering questing -1
# Read the file in data without first two rows as it is of no use.
marketing_data = pd.read_csv("marketing_analysis.csv",skiprows = 2)
#print the head of the data frame.
marketing_data.isnull().sum()
```

Out[3]:
```
customerid     0
age           20
salary         0
balance        0
marital        0
jobedu         0
targeted       0
default        0
housing        0
loan           0
contact        0
day            0
month         50
duration       0
campaign       0
pdays          0
previous       0
poutcome       0
response      30
dtype: int64
```

In [4]:
```
#Question 1: Implement three alternative approach(es) to handle missing values
#age and/or month columns. Explain why you decided to implement these approach
marketing_data['age']=marketing_data.age.replace(np.nan,'marketing_data.ag.mec
print('\nIn this way missing values of age column are replaced with the median
# Checking the missing values
marketing_data.isnull().sum()
```

Out[4]:
```
customerid     0
age            0
salary         0
balance        0
marital        0
jobedu         0
targeted       0
default        0
housing        0
loan           0
contact        0
day            0
month         50
duration       0
campaign       0
pdays          0
previous       0
poutcome       0
response      30
dtype: int64
```

In [5]:
```
#Again re-loading, checking the dataset for answering questing -1
# Read the file in data without first two rows as it is of no use.
marketing_data = pd.read_csv("marketing_analysis.csv",skiprows = 2)
#print the head of the data frame.
marketing_data.isnull().sum()
```

Out[5]:
```
customerid     0
age           20
salary         0
balance        0
marital        0
jobedu         0
targeted       0
default        0
housing        0
loan           0
contact        0
day            0
month         50
duration       0
campaign       0
pdays          0
previous       0
poutcome       0
response      30
dtype: int64
```

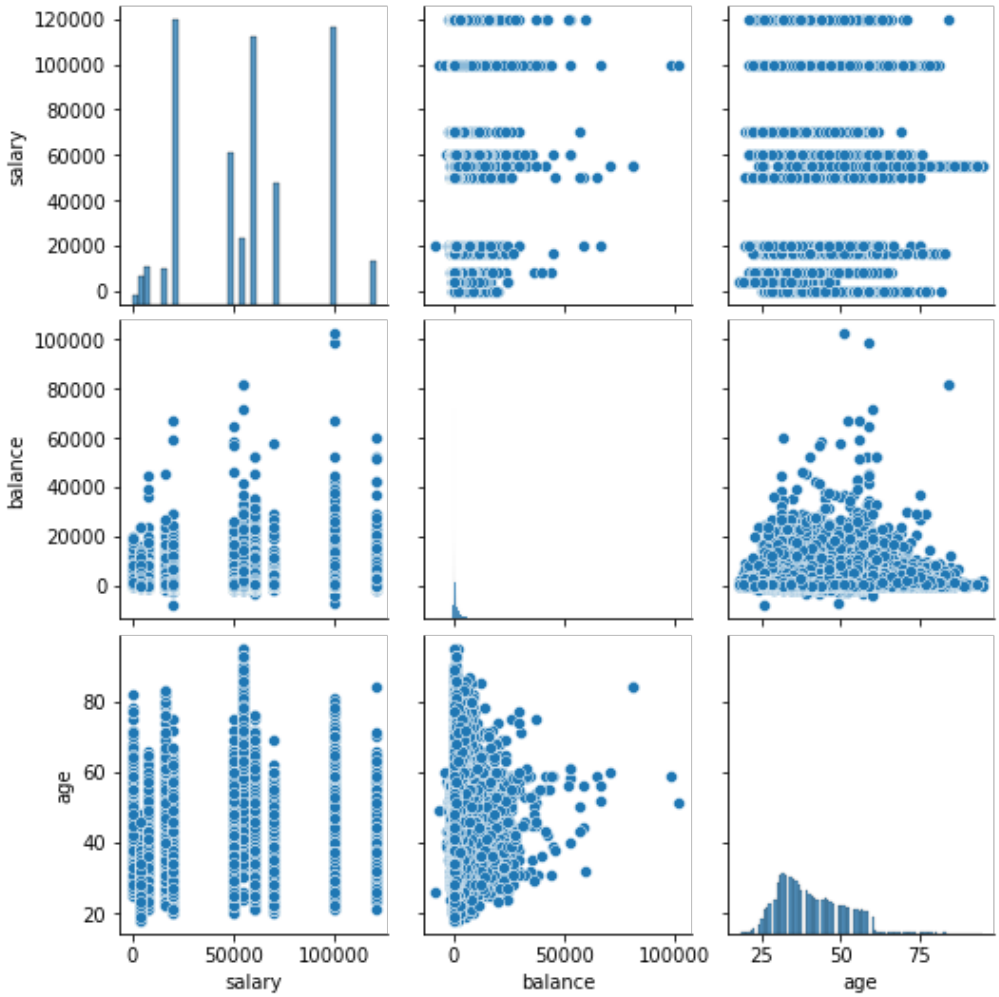
In [6]:
```
#Question 1: Implement three alternative approach(es) to handle missing values
#age and/or month columns. Explain why you decided to implement these approach
marketing_data = marketing_data.dropna()
print('\nI have used this approach to delete all the rows of all the columns i
marketing_data.isnull().sum()
```

```
I have used this approach to delete all the rows of all the columns includi
ng age column,
which have missing values so that missing values of the dataset can not cre
```

Out[6]:

```
customerid    0
age           0
salary        0
balance       0
marital       0
jobedu        0
targeted      0
default       0
housing       0
loan          0
contact       0
day           0
month         0
duration      0
campaign      0
pdays         0
previous      0
poutcome      0
response      0
dtype: int64
```

In [7]:
```
#Question 2: Is there outliers in our dataset? If yes, implement approach to h
#them. If no, demonstrate how you made this conclusion.
sns.pairplot(data=marketing_data, vars=['salary','balance','age'])
plt.show()
print('Yes, outliers are available in the columns and these can be handled by
print('----My Approach for The columns----')
print('For the age column, I would like to take it as it is as there is no men
print('Robust Scaling is useful for salary and balance columns because of many
```




```
Yes, outliers are available in the columns and these can be handled by norm
alizing
----My Approach for The columns----
For the age column, I would like to take it as it is as there is no mention
able outliers in the dataset.
The upper and lower bound of data are known and uniformally distributted.
Robust Scaling is useful for salary and balance columns because of many out
liers.
This approach has been implemented in the further coding.
Through this approach values of these columns are come to an similar scale
```

In [8]:
```
#Question 2: Is there outliers in our dataset? If yes, implement approach to h
#them. If no, demonstrate how you made this conclusion.(continue)
from sklearn.preprocessing import RobustScaler
# create a scaler object
scaler = RobustScaler()
```

```
# fit and transform the data
marketing_data['salary'] = scaler.fit_transform(marketing_data[['salary']])
marketing_data['balance'] = scaler.fit_transform(marketing_data[['balance']])
#marketing_data['age'] = scaler.fit_transform(marketing_data[['age']])
marketing_data
```

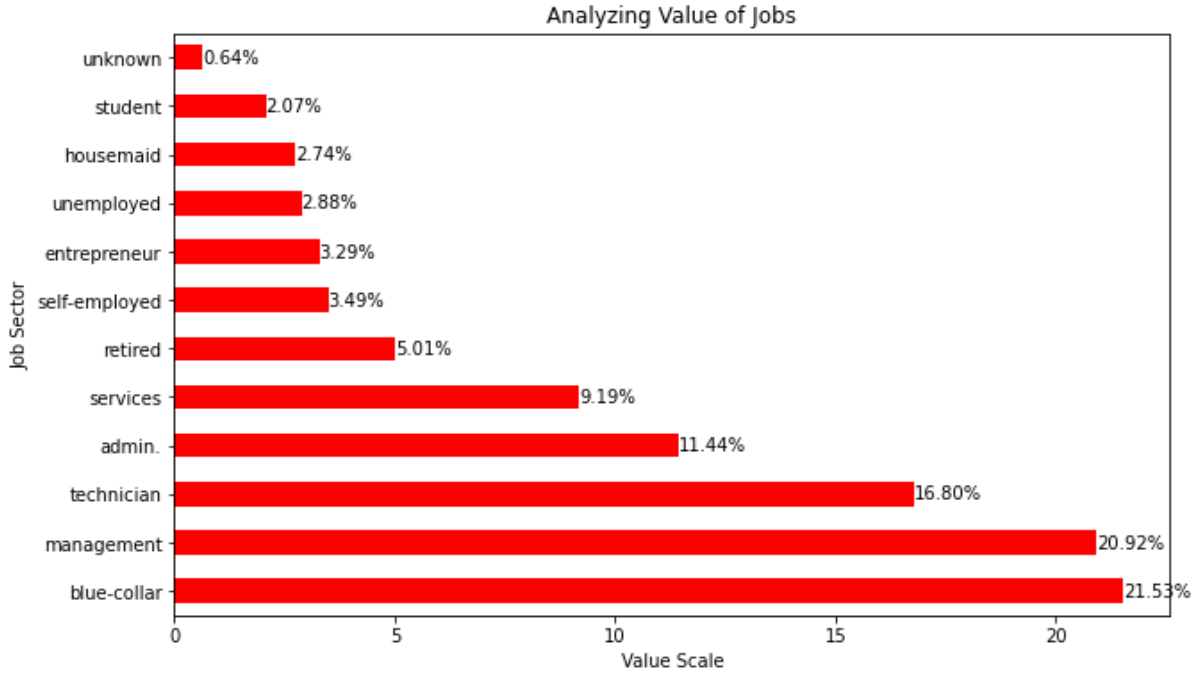
Out[8]:

	customerid	age	salary	balance	marital	jobedu	targeted	default	housing
0	1	58.0	0.8	1.250000	married	management,tertiary	yes	no	yes
1	2	44.0	0.0	-0.308997	single	technician,secondary	yes	no	yes
2	3	33.0	1.2	-0.328909	married	entrepreneur,secondary	yes	no	yes
3	4	47.0	-0.8	0.780236	married	blue-collar,unknown	no	no	yes
4	5	33.0	-1.2	-0.329646	single	unknown,unknown	no	no	no
...	...	...	...	...	...	...	...	...	...
45206	45207	51.0	0.0	0.278024	married	technician,tertiary	yes	no	no
45207	45208	71.0	-0.1	0.944690	divorced	retired,primary	yes	no	no
45208	45209	72.0	-0.1	3.884218	married	retired,secondary	yes	no	no
45209	45210	57.0	-0.8	0.162242	married	blue-collar,secondary	yes	no	no
45210	45211	37.0	1.2	1.860619	married	entrepreneur,secondary	yes	no	no

45111 rows × 19 columns

In [11]:
```
#Loading data again for further ans of question
marketing_data = pd.read_csv("marketing_analysis.csv",skiprows = 2)
marketing_data['job']= marketing_data["jobedu"].apply(lambda x:
x.split(",")[0])
marketing_data['education']= marketing_data["jobedu"].apply(lambda x:
x.split(",")[1])
# Drop the "jobedu" column from the dataframe.
marketing_data.drop('jobedu', axis = 1, inplace = True)
```

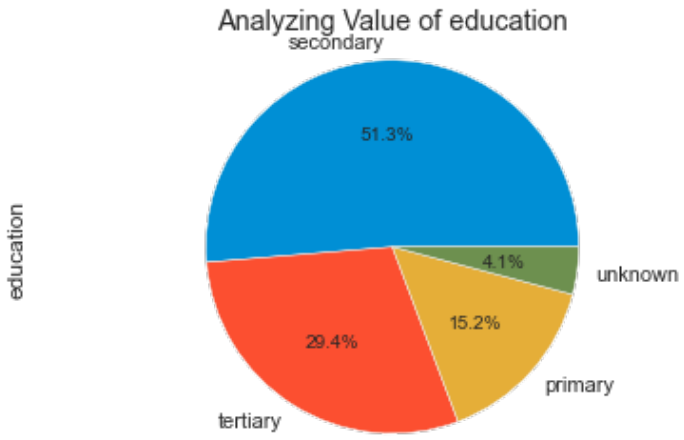
In [12]:
```
#Question 3: Add titles and data labels to this plot (Answered)
#plot the bar graph of percentage job categories
p1 = (marketing_data.job.value_counts(normalize=True)*100).plot.barh(figsize=(
plt.title('Analyzing Value of Jobs')
plt.xlabel('Value Scale')
plt.ylabel('Job Sector')
for p in p1.containers:
            p1.bar_label(p,fmt='%.2f%%', label_type='edge')
plt.show()
```



In [14]:
```
#Question 4: There are 'unknown' values in the dataset? Should we consider the
#missing? Explain how would you handle them, and demonstrate your solution. If
#decide to leave them as they are, explain why you made this decision.
print('----Explanation of My Answer-------\n')
print('It depends on the purpose of the study. For example, I want to measure

print('\nSteps of searching and droping the rows of unknown values')
print('step1-code for searching unknown values')

import string
unknownvalues = string.ascii_letters+string.punctuation
print('List of Unknown values: ', unknownvalues)
print('step2-code for Drop the rows: ')
# drop the rows
print('marketing_data.str.strip(unknownvalues).astype(bool).any()')
#print('This code is not working because no value is containing listed unknown
```

```
----Explanation of My Answer-------

It depends on the purpose of the study. For example, I want to measure the
economic condition of the people
of all ages from the given dataset. In this case, if the unknown values are
in the age, salary and balance
columns of the given dataset, I would like to drop all unknown values conta
ined rows. If I take these as it
is, I can not perform mathmetical calculations, which is necessary for the
analysis.
But, if such unknown values are stayed in the other columns, where I do not
have to do any numerical
calculation or if these columns do not have any influence in the analysis,
I will take these as it is because
this will not create any problem or add values in the analysis.
```

In [147]:
```
#Question 5: Add titles and data labels to this plot (Answered)
#plot the pie chart of education categories
marketing_data.education.value_counts(normalize=True).plot.pie(autopct='%1.1f%
plt.title('Analyzing Value of education')
#plt.pie(autopct='%1.1f%%')
plt.axis('equal')
plt.show()
```



In [15]:
```
# Loading data for further ans of the questing
#marketing_data.info()
```

In [20]:
```
#Question 6: Perform describe on all numerical columns in the dataset and comm
#on three most interesting observations.

print('----Observation about age---- \n')
print('-High difference between mean and std indicates data are more spread ou
print('-Dataset contains the data of adult people as the age starts from 18 in
print('-Dataset contains the data of young, middle aged and senior citizens.')
marketing_data.age.describe()
```

```
----Observation about age----

-High difference between mean and std indicates data are more spread out in
the column.
-Dataset contains the data of adult people as the age starts from 18 in the
dataset.
-Dataset contains the data of young, middle aged and senior citizens.
```

Out[20]:
```
count    45191.000000
mean        40.935651
std         10.619198
min         18.000000
25%         33.000000
50%         39.000000
75%         48.000000
max         95.000000
Name: age, dtype: float64
```

In [25]:
```
#Question 6: Perform describe on all numerical columns in the dataset and comm
#on three most interesting observations.
print('----Observation about salary---- \n')
print('-High difference between mean and std indicates data are more spread ou
print('-Minimum salary is 0 because many people in the dataset are not involve
print('-Maximum salary is 120000, which seems high compared to the average but
marketing_data.salary.describe()
```

```
----Observation about salary----

-High difference between mean and std indicates data are more spread out in
the column.
It is normal for this column as a very few people have high salary.
-Minimum salary is 0 because many people in the dataset are not involved wi
th job.
-Maximum salary is 120000, which seems high compared to the average but its
is normal because salary
 range varies extremely with the level of job.
```

Out[25]:
```
count     45211.000000
mean      57006.171065
std       32085.718415
min           0.000000
25%       20000.000000
50%       60000.000000
75%       70000.000000
max      120000.000000
Name: salary, dtype: float64
```

In [26]:
```
#Question 6: Perform describe on all numerical columns in the dataset and comm
```

```
#on three most interesting observations.
print('----Observation about balance---- \n')
print('-High difference between mean and std indicates data are more spread ou
print('-Minimum balance is minus 8019, which means some people have loans.')
print('-Maximum balance is 102127, which is quite high compared to the average
marketing_data.balance.describe()
```

```
----Observation about balance----

-High difference between mean and std indicates data are more spread out in
the column.
It is normal for this column as a very few people have high income source.
-Minimum balance is minus 8019, which means some people have loans.
-Maximum balance is 102127, which is quite high compared to the average 136
2.
This indicates that some people have extremly high balances
```

Out[26]:
```
count     45211.000000
mean       1362.272058
std        3044.765829
min       -8019.000000
25%          72.000000
50%         448.000000
75%        1428.000000
max      102127.000000
Name: balance, dtype: float64
```

In [29]:
```
#Question 6: Perform describe on all numerical columns in the dataset and comm
#on three most interesting observations.
print('----Observation about day---- \n')
print('-Minimum visiting days of customers is 1')
print('-Minimum visiting days of customers is 31')
print('-Average visiting days of customers is more than 8')
print('-Dataset contains new customers to most visiting customers as of the mi

marketing_data.day.describe()
```

```
----Observation about day----

-Minimum visiting days of customers is 1
-Minimum visiting days of customers is 31
-Average visiting days of customers is more than 8
-Dataset contains new customers to most visiting customers as of the minimu
m, average and maximum value.
```

Out[29]:
```
count    45211.000000
mean        15.806419
std          8.322476
min          1.000000
25%          8.000000
50%         16.000000
75%         21.000000
max         31.000000
Name: day, dtype: float64
```

In [28]:
```
#Question 6: Perform describe on all numerical columns in the dataset and comm
#on three most interesting observations.
print('----Observation about campaign---- \n')
print('-Minimum campaign is 1')
```

```
print('-Maximum campaign is 63')
print('-Average campaign is 2.7')
print('-Minimum and maximim difference value is extreme, whereas the mean and
marketing_data.campaign.describe()
```

```
----Observation about campaign----

-Minimum campaign is 1
-Maximum campaign is 63
-Average campaign is 2.7
-Minimum and maximim difference value is extreme, whereas the mean and std
value is closed,
which indicates that this column has extreme values or outliers
```

Out[28]:
```
count    45211.000000
mean         2.763841
std          3.098021
min          1.000000
25%          1.000000
50%          2.000000
75%          3.000000
max         63.000000
Name: campaign, dtype: float64
```

In [30]:
```
#Question 6: Perform describe on all numerical columns in the dataset and comm
#on three most interesting observations.
print('----Observation about pdays---- \n')
print('-Minimum pdays is -1')
print('-Maximum pdays is 871')
print('-Average pdays is 40')
print('-std is 100 ')
print('-std and mean difference indicates that data are extremely discreted')
marketing_data.pdays.describe()
```

```
----Observation about pdays----

-Minimum pdays is -1
-Maximum pdays is 871
-Average pdays is 40
-std is 100
-std and mean difference indicates that data are extremely discreted
```

Out[30]:
```
count    45211.000000
mean        40.197828
std        100.128746
min         -1.000000
25%         -1.000000
50%         -1.000000
75%         -1.000000
max        871.000000
Name: pdays, dtype: float64
```

In [34]:
```
#Question 6: Perform describe on all numerical columns in the dataset and comm
#on three most interesting observations.
print('----Observation about previous---- \n')
print('-Minimum and maximim value difference is extreme')
print('-The mean and std value is quite closed,\nwhich indicates that this col
print('-Three quarantile contain 0 values, that means no data or negative valu
marketing_data.previous.describe()
```

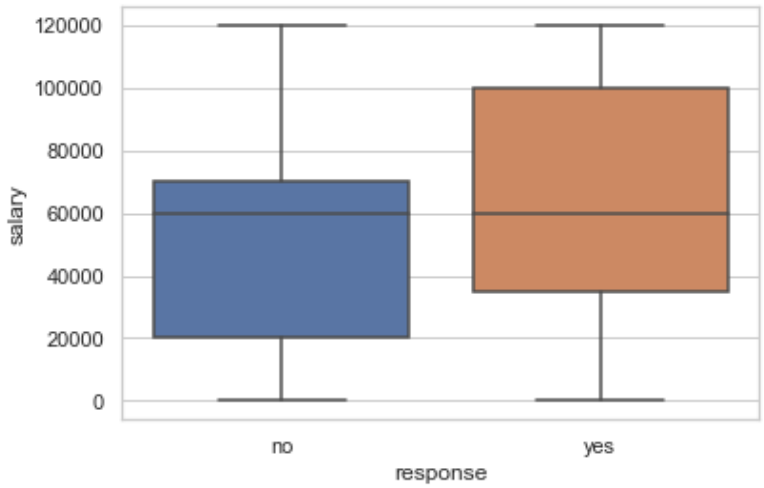
```
----Observation about previous----

-Minimum and maximim value difference is extreme
-The mean and std value is quite closed,
which indicates that this column has extreme values or outliers
-Three quarantile contain 0 values, that means no data or negative values
```

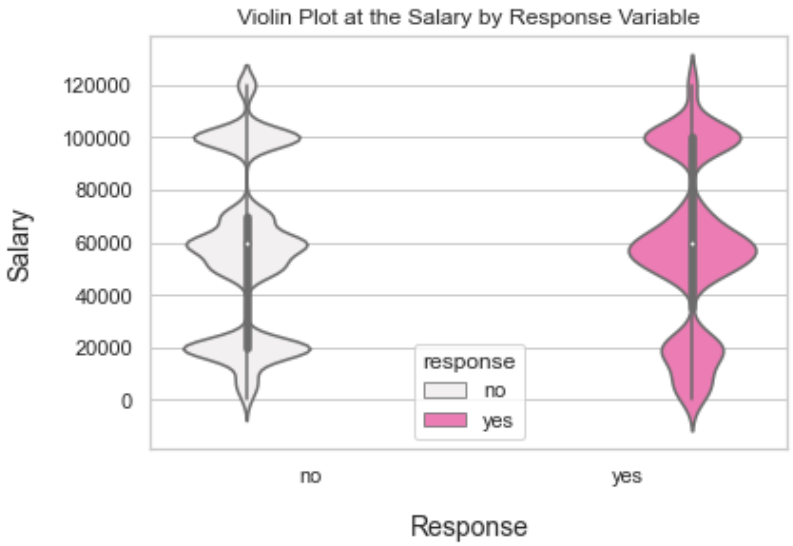
Out[34]:
```
count    45211.000000
mean         0.580323
std          2.303441
min          0.000000
25%          0.000000
50%          0.000000
75%          0.000000
max        275.000000
Name: previous, dtype: float64
```

In [35]:
```
#Question 7: Add plots formatting - axis titles and change dots formatting (co
#transparency) so it is easier to observe data patterns.(Answered)

#plot the scatter plot of balance and salary variable in data
plt.figure(figsize=(8, 5))
plt.scatter(marketing_data.salary, marketing_data.balance, color = 'hotpink',
plt.xlabel("Salary", fontsize=14, labelpad=15)
plt.ylabel("Balance", fontsize=14, labelpad=15)
plt.title('Data Distribution of Salary and Balance')
plt.show()
#plot the scatter plot of balance and age variable in data
#plt.figure(figsize=(8,7))
marketing_data.plot.scatter(x="age",y="balance", color = 'green', figsize=(8,6
plt.xlabel("Age", fontsize=14, labelpad=15)
plt.ylabel("Balance", fontsize=14, labelpad=15)
plt.title('Data Distribution of Age and Balance')
plt.show()
```

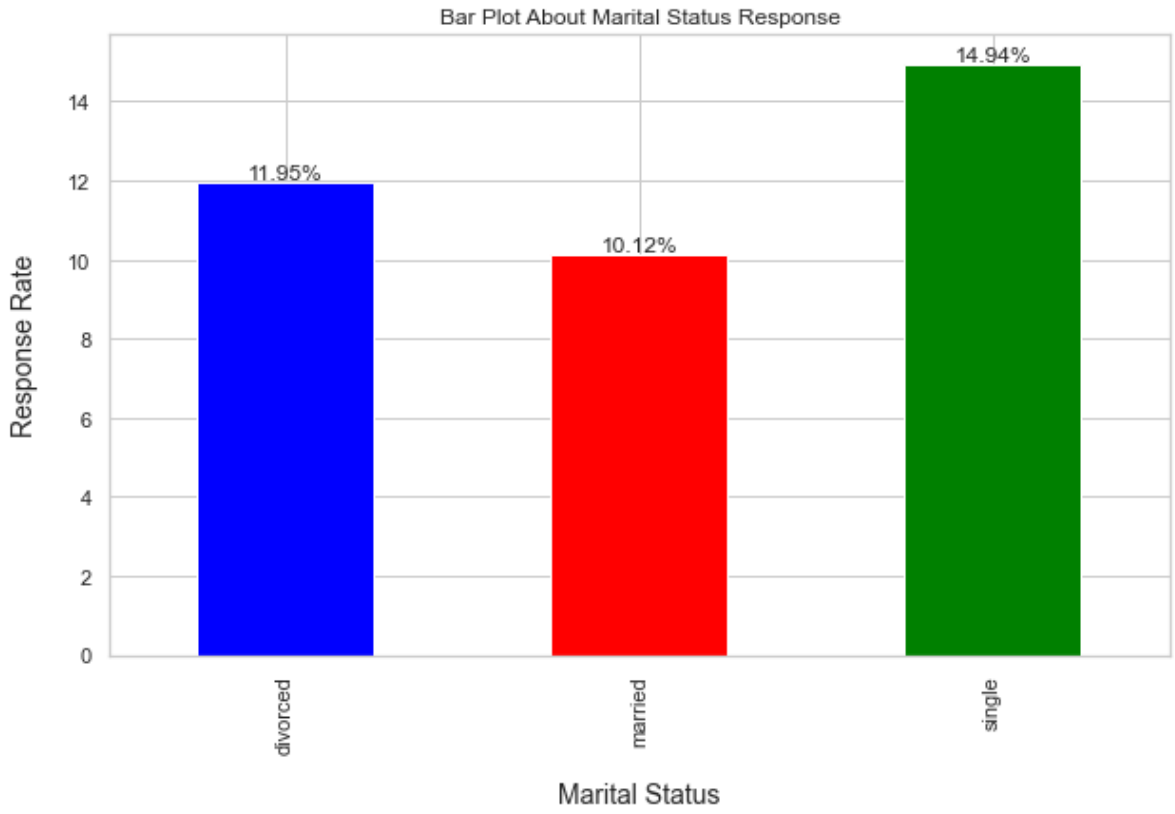
In [37]:
```
#Question 8: Seaborn library has variety of visualizations alternative to box
#aiming to analyze data distributio Implement violin plots instead of box pl
#at the salary by response variable. Change colors on the plot using one of th
#seaborn palettes.
#plot the box plot of salary for yes & no responses.
sns.boxplot(marketing_data.response, marketing_data.salary)
plt.show()

# Plotting violinplot
import seaborn as sns
sns.set_theme(style="whitegrid")
ax = sns.violinplot(x="response", y="salary", hue="response", color= 'hotpink'
plt.xlabel("Response", fontsize=14, labelpad=15)
plt.ylabel("Salary", fontsize=14, labelpad=15)
plt.title('Violin Plot at the Salary by Response Variable')
plt.show()
```

```
C:\Users\Taslima Akter\anaconda3\lib\site-packages\seaborn\_decorators.py:3
6: FutureWarning: Pass the following variables as keyword args: x, y. From
version 0.12, the only valid positional argument will be `data`, and passin
g other arguments without an explicit keyword will result in an error or mi
sinterpretation.
  warnings.warn(
```

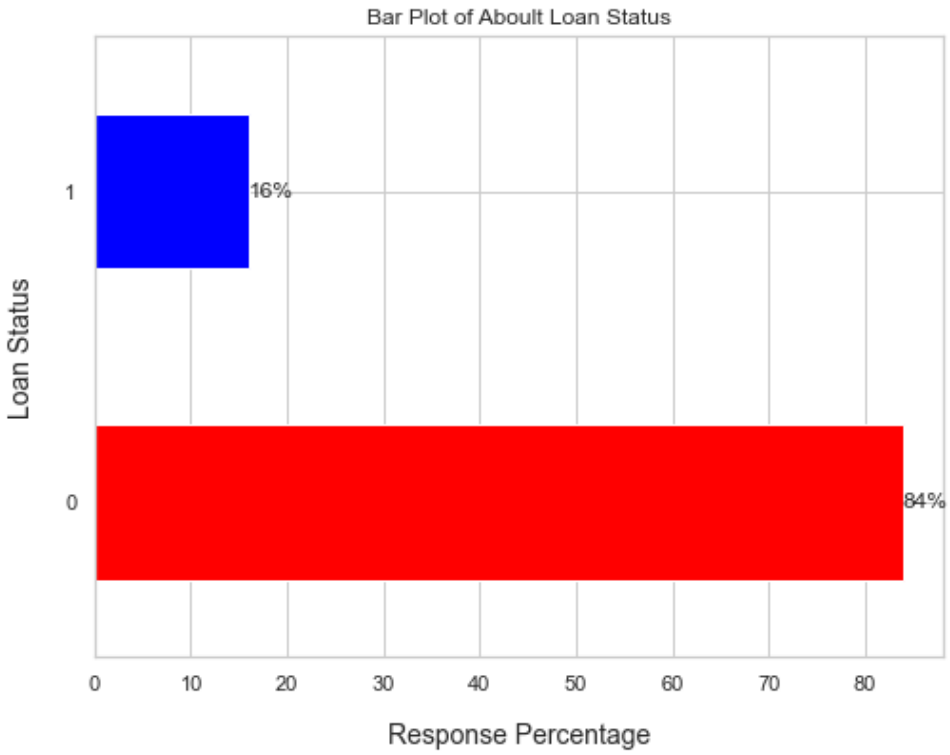




In [38]:
```
#Question 9: Add data labels and y-axis title and represent it as % (Answered)
marketing_data['response_rate']=np.where(marketing_data.response=='yes',1,0)

p1 = (marketing_data.groupby('marital')['response_rate'].mean()*100).plot.bar(
plt.xlabel("Marital Status", fontsize=14, labelpad=15)
plt.ylabel("Response Rate", fontsize=14, labelpad=15)
plt.title('Bar Plot About Marital Status Response')

for p in p1.containers:
    p1.bar_label(p,fmt='%.2f%%', label_type='edge')

plt.show()
```



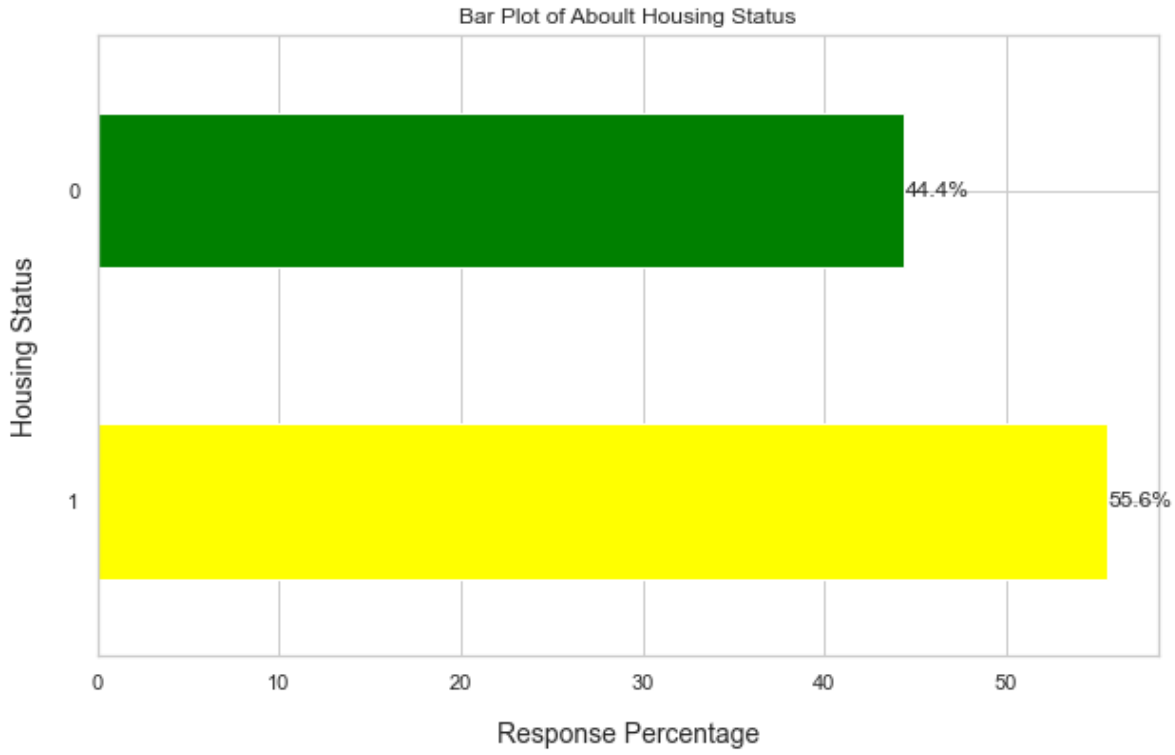
In [41]:
```
#Question 10: Similarly, plot the graphs for Loan vs Response rate, Housing Lo
#Response rate. Comment on your observations.
marketing_data['response_rate']=np.where(marketing_data.loan=='yes',1,0)
p1 = (marketing_data['response_rate'].value_counts(normalize=True)*100).plot.b
plt.xlabel("Response Percentage", fontsize=14, labelpad=15)
plt.ylabel("Loan Status", fontsize=14, labelpad=15)
plt.title('Bar Plot of Aboult Loan Status')
for p in p1.containers:
    p1.bar_label(p,fmt='%.0f%%')

plt.show()

print('---My observation---')
print('---Based on the response, 84% people of the dataset have no loan and 16
```




```
---My observation---
---Based on the response, 84% people of the dataset have no loan and 16% pe
ople have loan.
```

In [42]:
```
#Question 10: Similarly, plot the graphs for Loan vs Response rate, Housing Lo
#Response rate. Comment on your observations.

#Housing Loans vs Response rate
marketing_data['response_rate']=np.where(marketing_data.housing=='yes',1,0)
p1 = (marketing_data['response_rate'].value_counts(normalize=True)*100).plot.b
plt.xlabel("Response Percentage", fontsize=14, labelpad=15)
plt.ylabel("Housing Status", fontsize=14, labelpad=15)
plt.title('Bar Plot of Aboult Housing Status')
for p in p1.containers:
    p1.bar_label(p,fmt='%.1f%%')

plt.show()

print('---My observation---')
```

```
print('---Based on the response, 55.6% people of the dataset have housing loan
```




```
---My observation---
---Based on the response, 55.6% people of the dataset have housing loan and
44.4% people have no housing loan.
```

In [44]:
```
#Question 11: Similarly, plot the graphs for Job vs marital vs response Commer
#create pivot table
jobresult = pd.pivot_table(data=marketing_data, index='job',
columns='marital',values='response_rate')
print(jobresult)
```

marital	divorced	married	single
job			
admin.	0.668000	0.604530	0.609375
blue-collar	0.700000	0.731917	0.706554
entrepreneur	0.519553	0.596262	0.579832
housemaid	0.266304	0.324561	0.368056
management	0.522952	0.487407	0.497116
retired	0.270588	0.205084	0.194444
self-employed	0.557143	0.489426	0.450673
services	0.673953	0.671375	0.651085
student	0.166667	0.425926	0.256264
technician	0.557838	0.543189	0.533588
unemployed	0.421053	0.455540	0.344140
unknown	0.352941	0.064039	0.102941

In [45]:
```
#Question 11: Similarly, plot the graphs for Job vs marital vs response Commer
#create pivot table
#create heat map of job vs marital vs response_rate
sns.heatmap(jobresult, annot=True, cmap = 'RdYlGn', center=0.117)
plt.show()

#Obserrvation
```

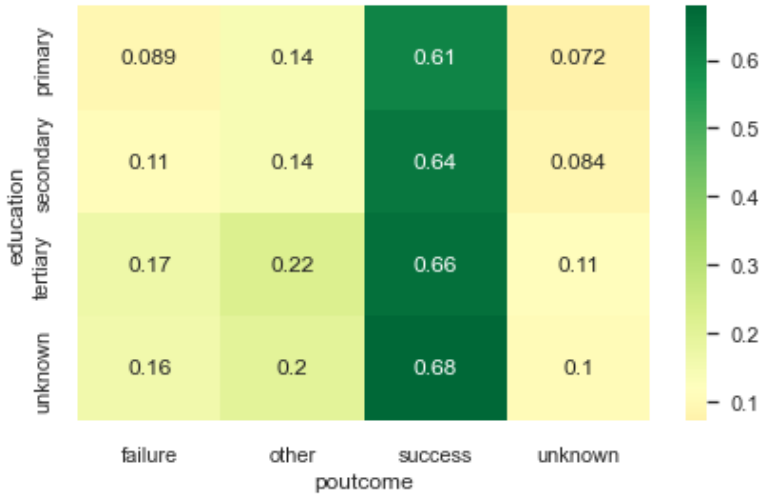
```
print('Divorce rate have positive corelation with the retirement-devorce rate
```


```
Divorce rate have positive corelation with the retirement-devorce rate is h
igher among retired couple,
meanwhile, married status is also have positive relation with retirement-ma
rried percentage is also higher
among the retired people, Student have positive corelation with single stat
us-Single rate is higher
among the students
```

In [46]:
```
#Question 11: Similarly, plot the graphs poutcome vs response. Comment on your
#poutcome vs response. Comment on your observations.
# Continuation of Question-11
#create pivot table
potresult = pd.pivot_table(data=marketing_data, index='education',
columns='poutcome',values='response_rate')
print(potresult)
```

```
poutcome     failure     other   success   unknown
education
primary     0.768274  0.663934  0.285714  0.548791
secondary   0.752414  0.679838  0.355556  0.589890
tertiary    0.621984  0.608456  0.281350  0.464531
unknown     0.564972  0.538462  0.222222  0.434811
```

In [168]:
```
#Question 11: Similarly, plot the graphs poutcome vs response. Comment on your
# Continuation of Question-11
#create pivot table
#create heat map of Education vs poutcome vs response_rate
sns.heatmap(potresult, annot=True, cmap = 'RdYlGn', center=0.117)
plt.show()

#Observvation
print('Failure is higher gradually the primary, secondary,\ntertiary and unkno
```




```
Failure is higher gradually the primary, secondary,
tertiary and unknown level or education as all level of education has posit
ive
corelation with the failure.Success rate is higher in the secondary level a
s it has positive corelation with success,
then primary,tertiary and unknown level of education comes, which also have
positive relation with success
```

Part-3-EDA Research

EDA- Exploratory data analysis is **a data exploration technique to understand the various aspects of the data**. It is a kind of summary of data. It is one of the most important steps before performing any machine learning or deep learning tasks.

Through Auto EDA we can have a lot of steps including some statistical tests, visualization of data using different kinds of plots by writing some codes.

Here I have used three Auto EDA libraries of Python:

1. **Pandas-Profiling:** an open-source python library that automates the EDA process and creates a detailed report. Used for large dataset profiling. It is fast and creates reports within a few seconds.
   The report contains: Overview of the dataset, variable properties, interaction of variables, correlation of variables, missing values and sample data.
2. **Sweetviz:** Open-source python auto-visualization library that generates a report, exploring the data with the help of high-density plots. It not only automates the EDA but is also used for comparing datasets and drawing inferences from it.
   The report contains: Overview of the dataset, variable properties, categorical associations, numerical associations, and most frequent, smallest, largest values for numerical features
3. **Autoviz**- Open-source python auto visualization library that mainly focuses on visualizing the relationship of the data by generating different types of plot.
   The report contains: Overview of the dataset, pairwise scatter plot of continuous variables, distribution of categorical variables, heat-maps of continuous variables and average numerical variable by each categorical variable

Remarks- Three of these libraries are used for profiling in the coding section and attached. There are so many pages of pandas_profiling, I have attached some pages of them.

Reference-

1. https://www.analyticsvidhya.com/blog/2021/04/top-python-libraries-to-automate-exploratory-data-analysis-in-2021/
2. https://towardsdatascience.com/autoviz-automatically-visualize-any-dataset-75876a4eede4
3. Class Lecture

In [1]:
```
#Part-3
#Again loading and reshaping, checking the dataset for Answering the ques
#import the useful libraries.
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import scipy
from scipy import stats
marketing_data = pd.read_csv("marketing_analysis.csv",skiprows = 2)
marketing_data['job']= marketing_data["jobedu"].apply(lambda x:
x.split(",")[0])
marketing_data['education']= marketing_data["jobedu"].apply(lambda x:
x.split(",")[1])
marketing_data.drop('jobedu', axis = 1, inplace = True)
```

In [2]:
```
#Research Find Python library that performs EDA automatically and re-do E
#for dataset provided using that library.
#EDA through pandas_profiling

from pandas_profiling import ProfileReport

prof = ProfileReport(marketing_data)
prof.to_file(output_file='marketing_data.html')
prof
```

Summarize dataset: 100% 99/99 [00:14<00:00, 5.30it/s, Completed]

Generate report structure: 100% 1/1 [00:04<00:00, 4.02s/it]

Render HTML: 100% 1/1 [00:02<00:00, 2.56s/it]

Export report to file: 100% 1/1 [00:00<00:00, 35.37it/s]

Overview

Dataset statistics

Number of variables	20
Number of observations	45211
Missing cells	100
Missing cells (%)	< 0.1%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	6.9 MiB
Average record size in memory	160.0 B

Variable types

Out[2]:

Variable types

Numeric	8
Categorical	7
Boolean	5

Alerts

duration has a high cardinality: 2646 distinct values	High cardinality
pdays is highly correlated with previous	High correlation
previous is highly correlated with pdays	High correlation

In [3]:
```
#Research Find Python library that performs EDA automatically and re-do E
#for dataset provided using that library.(Continue)
# EDA through sweetviz
import sweetviz as sv

sweetviz_report = sv.analyze(marketing_data)
sweetviz_report.show_html('marketing_data_SWB.html')
```

Done! Use 'show' commands to display/save. [100%] 00:00 -> (00:00 left)

```
Report marketing_data_SWB.html was generated! NOTEBOOK/COLAB USERS: th
e web browser MAY not pop up, regardless, the report IS saved in your
notebook/colab files.
```

In [4]:
```
#Research Find Python library that performs EDA automatically and re-do E
#for dataset provided using that library.(Continue)
# EDA through sweetviz Comparison
sweetviz_compare = sv.compare(marketing_data.loc[marketing_data['marital'
sweetviz_compare
```

Done! Use 'show' commands to display/save. [100%] 00:01 -> (00:00 left)

Out[4]: <sweetviz.dataframe_report.DataframeReport at 0x14fa8e912b0>

In [5]:
```
#Research Find Python library that performs EDA automatically and re-do E
#for dataset provided using that library.(Continue)
# EDA through sweetviz Comparison
sweetviz_compare.show_html('marketing_data_marital_compare.html')
```

```
Report marketing_data_marital_compare.html was generated! NOTEBOOK/COL
AB USERS: the web browser MAY not pop up, regardless, the report IS sa
ved in your notebook/colab files.
```

In [6]:
```
#Research Find Python library that performs EDA automatically and re-do E
#for dataset provided using that library.(Continue)
## EDA through autoviz
from autoviz.AutoViz_Class import AutoViz_Class

#instantiate the Autoviz class
AV = AutoViz_Class()
marketing_data_vis = AV.AutoViz('marketing_analysis.csv', verbose = 2 )
```

```
Alert! from version 0.1.36. After importing, you must do '%matplotlib
inline' to display charts in Jupyter Notebooks.
    AV = AutoViz_Class()
    AV.AutoViz(filename, sep=',', depVar='', dfte=None, header=0, verb
ose=0, lowess=False,
               chart_format='svg',max_rows_analyzed=150000,max_cols_an
alyzed=30, save_plot_dir=None)
```

```
Shape of your Data Set loaded: (45213, 19)
############## C L A S S I F Y I N G  V A R I A B L E S  ############
#######
Classifying variables in data set...
Data Set Shape: 45213 rows, 19 cols
Data Set columns info:
* banking marketing: 0 nulls, 45213 unique vals, most common: {'custom
er id and age.': 1, '30200': 1}
* Unnamed: 1: 21 nulls, 145 unique vals, most common: {'32': 1509, '31
': 1487}
* Unnamed: 2: 0 nulls, 24 unique vals, most common: {'20000': 7290, '6
0000': 7036}
* Unnamed: 3: 1 nulls, 10335 unique vals, most common: {'0': 2767, 0:
747}
* Unnamed: 4: 0 nulls, 5 unique vals, most common: {'married': 27214,
'single': 12790}
* Unnamed: 5: 1 nulls, 49 unique vals, most common: {'management,terti
ary': 7801, 'blue-collar,secondary': 5371}
* Unnamed: 6: 0 nulls, 4 unique vals, most common: {'yes': 37091, 'no
': 8120}
* Unnamed: 7: 1 nulls, 3 unique vals, most common: {'no': 44396, 'yes
': 815}
* Unnamed: 8: 0 nulls, 4 unique vals, most common: {'yes': 25130, 'no
': 20081}
* Unnamed: 9: 1 nulls, 3 unique vals, most common: {'no': 37967, 'yes
': 7244}
* Unnamed: 10: 0 nulls, 5 unique vals, most common: {'cellular': 2928
5, 'unknown': 13020}
* Unnamed: 11: 1 nulls, 63 unique vals, most common: {'20': 2030, '21
': 1880}
* Unnamed: 12: 50 nulls, 14 unique vals, most common: {'may, 2017': 13
747, 'jul, 2017': 6888}
* Unnamed: 13: 0 nulls, 2648 unique vals, most common: {'1.5 min': 13
8, '2.06666666666667 min': 129}
* Unnamed: 14: 1 nulls, 67 unique vals, most common: {'1': 11502, '2':
9248}
* Unnamed: 15: 1 nulls, 837 unique vals, most common: {'-1': 30277, -
1: 6677}
* Unnamed: 16: 1 nulls, 72 unique vals, most common: {'0': 30277, 0: 6
677}
* Unnamed: 17: 0 nulls, 6 unique vals, most common: {'unknown': 36959,
'failure': 4901}
* Unnamed: 18: 30 nulls, 4 unique vals, most common: {'no': 39894, 'ye
s': 5287}
-------------------------------------------------------------------
```

```
    Numeric Columns: []
    Integer-Categorical Columns: []
    String-Categorical Columns: ['Unnamed: 2', 'Unnamed: 4', 'Unnamed:
6', 'Unnamed: 7', 'Unnamed: 8', 'Unnamed: 9', 'Unnamed: 10', 'Unnamed:
12', 'Unnamed: 17', 'Unnamed: 18']
    Factor-Categorical Columns: []
    String-Boolean Columns: []
    Numeric-Boolean Columns: []
    Discrete String Columns: ['Unnamed: 1', 'Unnamed: 3', 'Unnamed: 5
', 'Unnamed: 11', 'Unnamed: 13', 'Unnamed: 14', 'Unnamed: 15', 'Unname
d: 16']
    NLP text Columns: []
    Date Time Columns: []
    ID Columns: ['banking marketing']
    Columns that will not be considered in modeling: []
    19 Predictors classified...
        1 variables removed since they were ID or low-information vari
ables
    List of variables removed: ['banking marketing']
No continuous variables in this data set. No visualization can be perf
ormed
Not able to read or load file. Please check your inputs and try agai
n
```

Overview

Dataset statistics

Number of variables	20
Number of observations	45211
Missing cells	100
Missing cells (%)	< 0.1%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	6.9 MiB
Average record size in memory	160.0 B

Variable types

Numeric	8
Categorical	7
Boolean	5

Alerts

`duration` has a high cardinality: 2646 distinct values	High cardinality
`pdays` is highly correlated with `previous`	High correlation
`previous` is highly correlated with `pdays`	High correlation
`pdays` is highly correlated with `previous`	High correlation
`previous` is highly correlated with `pdays`	High correlation
`month` is highly correlated with `housing` and 1 other fields (housing, contact)	High correlation
`housing` is highly correlated with `month`	High correlation
`targeted` is highly correlated with `education`	High correlation
`education` is highly correlated with `targeted`	High correlation
`contact` is highly correlated with `month`	High correlation
`customerid` is highly correlated with `housing` and 6 other fields (housing, contact, day, month, pdays, poutcome, response)	High correlation
`age` is highly correlated with `job`	High correlation
`salary` is highly correlated with `job` and 1 other fields (job, education)	High correlation
`targeted` is highly correlated with `education`	High correlation
`housing` is highly correlated with `customerid` and 1 other fields (customerid, month)	High correlation
`contact` is highly correlated with `customerid` and 1 other fields (customerid, month)	High correlation
`day` is highly correlated with `customerid` and 1 other fields (customerid, month)	High correlation
`month` is highly correlated with `customerid` and 3 other fields (customerid, housing, contact, day)	High correlation
`pdays` is highly correlated with `customerid` and 1 other fields (customerid, poutcome)	High correlation
`poutcome` is highly correlated with `customerid` and 1 other fields (customerid, pdays)	High correlation
`response` is highly correlated with `customerid`	High correlation
`job` is highly correlated with `age` and 2 other fields (age, salary, education)	High correlation
`education` is highly correlated with `salary` and 2 other fields (salary, targeted, job)	High correlation
`previous` is highly skewed (γ1 = 41.84645447)	Skewed
`customerid` is uniformly distributed	Uniform
`customerid` has unique values	Unique
`balance` has 3514 (7.8%) zeros	Zeros
`previous` has 36954 (81.7%) zeros	Zeros

Reproduction

Analysis started	2022-03-23 01:45:57.906788
Analysis finished	2022-03-23 01:46:11.984845
Duration	14.08 seconds
Software version	pandas-profiling v3.1.0 (https://github.com/pandas-profiling/pandas-profiling)

Download configuration

config.json (data:text/plain;charset=utf-8,%7B%22title%22%3A%20%22Pandas%20Profiling%20Report%22%2C%20%22dataset%22%3A%20%7B%22description%22%3A%20%22%22%2C%20%22creator%22%3A%20%22%22%2C%20%22author%22%3A%20%22%22%2C%20%22copyright_holder%22%3A%20%22%22%2C%20%22copyright_year%22%3A%20%22%22%2C%20%22url%22%3A%20%22%22%7D%2C%20%22variables%22%3A%20%7B%22descriptions%22%3A%20%7B%7D%7D%2C%20%22infer_dtypes%22%3A%20true%2C%20%22show_variable_description%22%3A%20true%2C%20%22pool_size%22%3A%200%2C%20%22progress_bar%22%3A%20true%2C%20%22vars%22%3A%20%7B%22num%22%3A%20%7B%22quantiles%22%3A%20%5B0.05%2C%200.25%2C%200.5%2C%200.75%2C%200.95%5D%2C%20%22skewness_threshold%22%3A%2020%2C%20%22low_categorical_threshold%22%3A%205%2C%20%22chi_squared_threshold%22%3A%200.999%7D%2C%20%22cat%22%3A%20%7B%22length%22%3A%20true%2C%20%22characters%22%3A%20true%2C%20%22words%22%3A%20true%2C%20%22cardinality_threshold%22%3A%2050%2C%20%22n_obs%22%3A%205%2C%20%22chi_squared_threshold%22%3A%200.999%2C%20%22coerce_str_to_date%22%3A%20false%2C%20%22redact%22%3A%20false%2C%20%22histogram_largest%22%3A%2050%7D%2C%20%22image%22%3A%20%7B%22active%22%3A%20false%2C%20%22exif%22%3A%20true%2C%20%22hash%22%3A%20true%7D%2C%20%22bool%22%3A%20%7B%22n_obs%22%3A%203%2C%20%22mappings%22%3A%20%7B%22t%22%3A%20true%2C%20%22f%22%3A%20false%2C%20%22yes%22%3A%20true%2C%20%22no%22%3A%20false%2C%20%22y%22%3A%20true%2C%20%22n%22%3A%20false%2C%20%22true%22%3A%20true%2C%20%22false%22%3A%20false%7D%7D%2C%20%22path%22%3A%20%7B%22active%22%3A%20false%7D%2C%20%22file%22%3A%20%7B%22active%22%3A%20false%7D%2C%20%22url%22%3A%20%7B%22active%22%3A%20false%7D%7D%2C%20%22sort%22%3A%20null%2C%20%22missing_diagrams%22%3A%20%7B%22bar%22%3A%20true%2C%20%22matrix%22%3A%20true%2C%20%22dendrogram%22%3A%20true%2C%20%22heatmap%22%3A%20true%7D%2C%20%22correlations%22%3A%20%7B%22spearman%22%3A%20%7B%22key%22%3A%20%22spearman%22%2C%20%22calculate%22%3A%20true%2C%20%22warn_high_correlations%22%3A%2010%2C%20%22threshold%22%3A%200.5%7D%2C%20%22pearson%22%3A%20%7B%22key%22%3A%20%22pearson%22%2C%20%22calculate%22%3A%20true%2C%20%22warn_high_correlations%22%3A%2010%2C%20%22threshold%22%3A%200.5%7D%2C%20%22kendall%22%3A%20%7B%22key%22%3A%20%22kendall%22%2C%20%22calculate%22%3A%20true%2C%20%22warn_high_correlations%22%3A%2010%2C%20%22threshold%22%3A%200.5%7D%2C%20%22cramers%22%3A%20%7B%22key%22%3A%20%22cramers%22%2C%20%22calculate%22%3A%20true%2C%20%22warn_high_correlations%22%3A%2010%2C%20%22threshold%22%3A%200.5%7D%2C%20%22phi_k%22%3A%20%7B%22key%22%3A%20%22phi_k%22%2C%20%22calculate%22%3A%20true%2C%20%22warn_high_correlations%22%3A%2010%2C%20%22threshold%22%3A%200.5%7D%7D%2C%20

%22interactions%22%3A%20%7B%22continuous%22%3A%20true%2C%20
%22targets%22%3A%20%5B%5D%7D%2C%20
%22categorical_maximum_correlation_distinct%22%3A%20100%2C%20
%22memory_deep%22%3A%20false%2C%20%22plot%22%3A%20%7B
%22missing%22%3A%20%7B%22force_labels%22%3A%20true%2C%20
%22cmap%22%3A%20%22RdBu%22%7D%2C%20%22image_format%22%3A
%20%22svg%22%2C%20%22correlation%22%3A%20%7B%22cmap%22%3A
%20%22RdBu%22%2C%20%22bad%22%3A%20%22%23000000%22%7D
%2C%20%22dpi%22%3A%20800%2C%20%22histogram%22%3A%20%7B
%22bins%22%3A%2050%2C%20%22max_bins%22%3A%20250%2C%20
%22x_axis_labels%22%3A%20true%7D%2C%20%22scatter_threshold%22%3A
%201000%2C%20%22pie%22%3A%20%7B%22max_unique%22%3A%2010%7D
%7D%2C%20%22duplicates%22%3A%20%7B%22head%22%3A%2010%2C
%20%22key%22%3A%20%22%23%20duplicates%22%7D%2C%20%22samples
%22%3A%20%7B%22head%22%3A%2010%2C%20%22tail%22%3A%2010%2C
%20%22random%22%3A%200%7D%2C%20%22reject_variables%22%3A%20true
%2C%20%22n_obs_unique%22%3A%2010%2C%20%22n_freq_table_max
%22%3A%2010%2C%20%22n_extreme_obs%22%3A%2010%2C%20%22report
%22%3A%20%7B%22precision%22%3A%2010%7D%2C%20%22html%22%3A
%20%7B%22style%22%3A%20%7B%22primary_color%22%3A%20%22
%23337ab7%22%2C%20%22logo%22%3A%20%22%22%2C%20%22theme
%22%3A%20null%7D%2C%20%22navbar_show%22%3A%20true%2C%20
%22minify_html%22%3A%20true%2C%20%22use_local_assets%22%3A%20true
%2C%20%22inline%22%3A%20true%2C%20%22assets_prefix%22%3A%20null
%2C%20%22assets_path%22%3A%20null%2C%20%22full_width%22%3A
%20false%7D%2C%20%22notebook%22%3A%20%7B%22iframe%22%3A
%20%7B%22height%22%3A%20%22800px%22%2C%20%22width%22%3A
%20%22100%25%22%2C%20%22attribute%22%3A%20%22srcdoc%22%7D
%7D%7D)

Variables

customerid

Real number ($\mathbb{R}_{\geq 0}$)

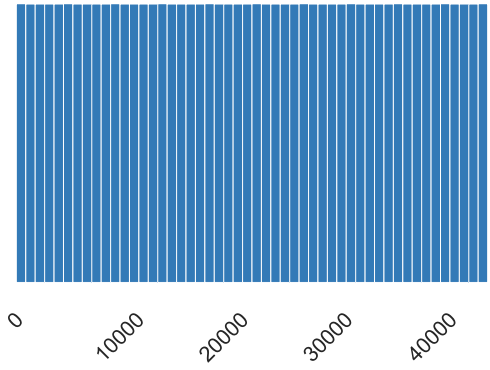
HIGH CORRELATION (This variable has a high correlation with 7 fields: housing, contact, day, month, pdays, poutcome, response)

UNIFORM

UNIQUE

Distinct	45211
Distinct (%)	100.0%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%
Mean	22606

Minimum	1
Maximum	45211
Zeros	0
Zeros (%)	0.0%
Negative	0
Negative (%)	0.0%
Memory size	353.3 KiB

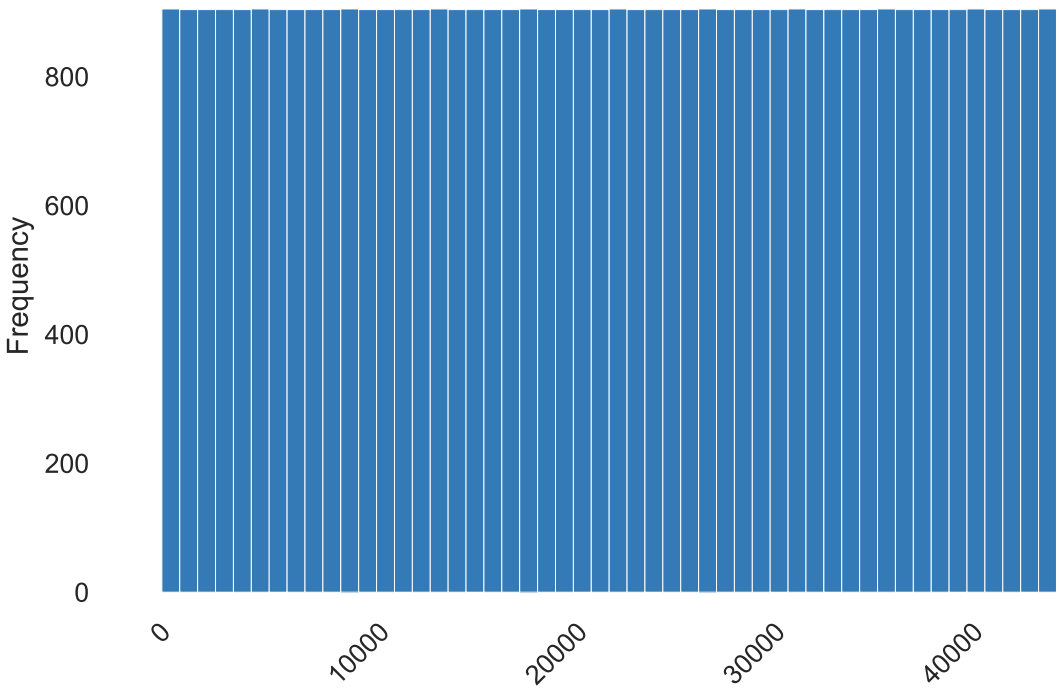


Quantile statistics

Minimum	1
5-th percentile	2261.5
Q1	11303.5
median	22606
Q3	33908.5
95-th percentile	42950.5
Maximum	45211
Range	45210
Interquartile range (IQR)	22605

Descriptive statistics

Standard deviation	13051.43585
Coefficient of variation (CV)	0.5773438842
Kurtosis	-1.2
Mean	22606
Median Absolute Deviation (MAD)	11303
Skewness	0
Sum	1022039866
Variance	170339977.7
Monotonicity	Strictly increasing



Histogram with fixed size bins (bins=50)

Value	Count	Frequency (%)
1	1	< 0.1%
30200	1	< 0.1%
30136	1	< 0.1%
30137	1	< 0.1%
30138	1	< 0.1%
30139	1	< 0.1%
30140	1	< 0.1%
30141	1	< 0.1%
30142	1	< 0.1%
30143	1	< 0.1%
Other values (45201)	45201	> 99.9%

Value	Count	Frequency (%)
1	1	< 0.1%
2	1	< 0.1%
3	1	< 0.1%
4	1	< 0.1%
5	1	< 0.1%
6	1	< 0.1%
7	1	< 0.1%
8	1	< 0.1%
9	1	< 0.1%
10	1	< 0.1%

Value	Count	Frequency (%)
45211	1	< 0.1%
45210	1	< 0.1%
45209	1	< 0.1%
45208	1	< 0.1%
45207	1	< 0.1%
45206	1	< 0.1%
45205	1	< 0.1%
45204	1	< 0.1%
45203	1	< 0.1%
45202	1	< 0.1%

Value	Count	Frequency (%)
95	2	< 0.1%
94	1	< 0.1%
93	2	< 0.1%
92	2	< 0.1%
90	2	< 0.1%
89	3	< 0.1%
88	2	< 0.1%
87	4	< 0.1%
86	9	< 0.1%
85	5	< 0.1%

salary

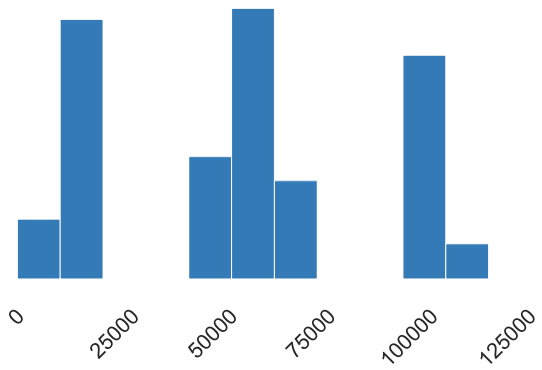
Real number ($\mathbb{R}_{\geq 0}$)

HIGH

CORRELATION (This variable has a high correlation with 2 fields: job, education)

Distinct	11
Distinct (%)	< 0.1%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%
Mean	57006.17106

Minimum	0
Maximum	120000
Zeros	288
Zeros (%)	0.6%
Negative	0
Negative (%)	0.0%
Memory size	353.3 KiB



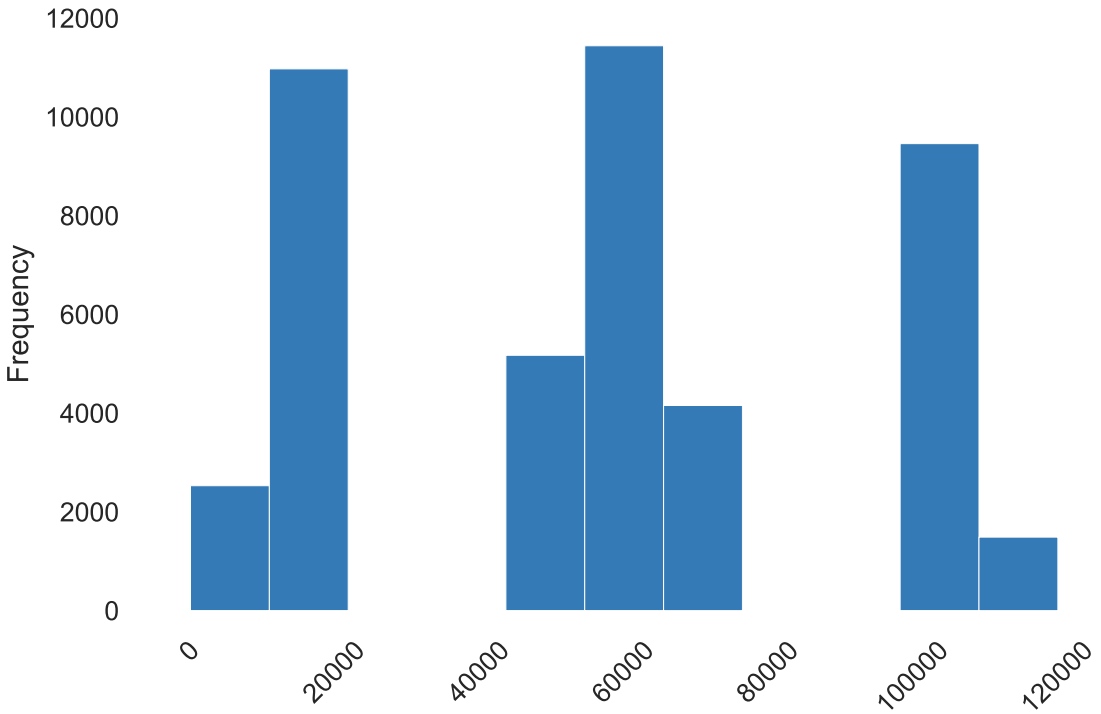
Quantile statistics

Minimum	0

5-th percentile	8000
Q1	20000
median	60000
Q3	70000
95-th percentile	100000
Maximum	120000
Range	120000
Interquartile range (IQR)	50000

Descriptive statistics

Standard deviation	32085.71842
Coefficient of variation (CV)	0.5628464045
Kurtosis	-1.005143279
Mean	57006.17106
Median Absolute Deviation (MAD)	40000
Skewness	0.1378290938
Sum	2577306000
Variance	1029493326
Monotonicity	Not monotonic



Histogram with fixed size bins (bins=11)

Value	Count	Frequency (%)
20000	9732	21.5%
100000	9458	20.9%
60000	9176	20.3%
50000	5171	11.4%
70000	4154	9.2%
55000	2264	5.0%
120000	1487	3.3%
8000	1303	2.9%
16000	1240	2.7%
4000	938	2.1%

Sweetviz
2.1.3

Get updates, docs & report issues here

Created & maintained by Francois Bertrand
Graphic design by Jean-Francois Hains

DataFrame

45211	ROWS
0	DUPLICATES
37.1 MB	RAM
20	FEATURES
11	CATEGORICAL
8	NUMERICAL
1	TEXT

ASSOCIATIONS

DataFrame

1 customerid

VALUES:	45,211 (100%)	MAX	45,211	RANGE	45,210
MISSING:	---	95%	42,950	IQR	22,605
		Q3	33,908	STD	13,051
DISTINCT:	45,211 (100%)	MEDIAN	22,606	VAR	170.3M
		AVG	22,606		
ZEROES:	---	Q1	11,304	KURT.	-1.20
		5%	2,262	SKEW	0.00
		MIN	1	SUM	1.0B

2 age

VALUES:	45,191 (>99%)	MAX	95.0	RANGE	77.0
MISSING:	20 (<1%)	95%	59.0	IQR	15.0
		Q3	48.0	STD	10.6
DISTINCT:	77 (<1%)	AVG	40.9	VAR	113
		MEDIAN	39.0		
ZEROES:	---	Q1	33.0	KURT.	0.320
		5%	27.0	SKEW	0.685
		MIN	18.0	SUM	1.8M

3 salary

VALUES:	45,211 (100%)	MAX	120k	RANGE	120k
MISSING:	---	95%	100k	IQR	50,000
		Q3	70k	STD	32,086
DISTINCT:	11 (<1%)	MEDIAN	60k	VAR	1.0B
		AVG	57k		
ZEROES:	288 (<1%)	Q1	20k	KURT.	-1.01
		5%	8k	SKEW	0.138
		MIN	0k	SUM	2.6B

4 balance

VALUES:	45,211 (100%)	MAX	102k	RANGE	110k
MISSING:	---	95%	6k	IQR	1,356
		Q3	1k	STD	3,045
DISTINCT:	7,168 (16%)	AVG	1k	VAR	9.3M
		MEDIAN	0k		
ZEROES:	3,514 (8%)	Q1	0k	KURT.	141
		5%	-0k	SKEW	8.36
		MIN	-8k	SUM	61.6M

5 marital

VALUES:	45,211 (100%)
MISSING:	---
DISTINCT:	3 (<1%)

6 **targeted**

VALUES: 45,211 (100%)
MISSING: ---

DISTINCT: 2 (<1%)

7 **default**

VALUES: 45,211 (100%)
MISSING: ---

DISTINCT: 2 (<1%)

8 **housing**

VALUES: 45,211 (100%)
MISSING: ---

DISTINCT: 2 (<1%)

9 **loan**

VALUES: 45,211 (100%)
MISSING: ---

DISTINCT: 2 (<1%)

10 **contact**

VALUES: 45,211 (100%)
MISSING: ---

DISTINCT: 3 (<1%)

unk

11 **day**

VALUES: 45,211 (100%)
MISSING: ---

MAX	31.0	RANGE	30.0
95%	29.0	IQR	13.0
Q3	21.0	STD	8.32

12 month

VALUES:	45,161	(>99%)
MISSING:	50	(<1%)
DISTINCT:	12	(<1%)

may,
jul,
aug,
jun,
(

13 duration

VALUES:	45,211	(100%)
MISSING:	---	
DISTINCT:	2,646	(6%)

138	<1%	1.5 min
129	<1%	1.68333333333333 min
129	<1%	2.06666666666667 min
127	<1%	1.73333333333333 min
127	<1%	2.03333333333333 min
126	<1%	1.33333333333333 min
125	<1%	1.46666666666667 min
44,310	98%	(Other)

14 campaign

VALUES:	45,211	(100%)
MISSING:	---	
DISTINCT:	48	(<1%)
ZEROES:	---	

MAX	63.0		RANGE	62.0
95%	8.0		IQR	2.00
Q3	3.0		STD	3.10
AVG	2.8		VAR	9.60
MEDIAN	2.0			
Q1	1.0		KURT.	39.2
5%	1.0		SKEW	4.90
MIN	1.0		SUM	125k

15 pdays

VALUES:	45,211	(100%)
MISSING:	---	
DISTINCT:	559	(1%)
ZEROES:	---	

MAX	871		RANGE	872
95%	317		IQR	0.00
Q3	-1		STD	100
AVG	40		VAR	10,026
MEDIAN	-1			
Q1	-1		KURT.	6.94
5%	-1		SKEW	2.62
MIN	-1		SUM	1.8M

16 previous

VALUES:	45,211	(100%)
MISSING:	---	
DISTINCT:	41	(<1%)
ZEROES:	36,954	(82%)

MAX	275		RANGE	275
95%	3		IQR	0.00
Q3	0		STD	2.30
AVG	1		VAR	5.31
MEDIAN	0			
Q1	0		KURT.	4,507
5%	0		SKEW	41.8
MIN	0		SUM	26,237

17 poutcome

18 **response**

VALUES:	45,181	(>99%)
MISSING:	30	(<1%)
DISTINCT:	2	(<1%)

19 **job**

VALUES:	45,211	(100%)
MISSING:	---	
DISTINCT:	12	(<1%)

blue-
manage
techn
a
(

20 **education**

VALUES:	45,211	(100%)
MISSING:	---	
DISTINCT:	4	(<1%)

seco
te
pr
unk

Sweetviz
2.1.3

Get updates, docs & report issues here

Created & maintained by Francois Bertrand
Graphic design by Jean-Francois Hains

DataFrame

ASSOCIATIONS

12790	ROWS
0	DUPLICATES
10.6 MB	RAM
20	FEATURES
11	CATEGORICAL
8	NUMERICAL
1	TEXT

DataFrame

1 customerid

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	12,790 (100%)	17,997 (100%)
ZEROES:	---	---

MAX	45,206	45,208
95%	43,607	43,388
Q3	36,580	35,481
MEDIAN	26,810	25,108
AVG	24,553	23,743
Q1	12,557	11,871
5%	2,331	2,228
MIN	2	2

RANGE	45,204	45,206
IQR	24,022	23,610
STD	13,508	13,391
VAR	182.5M	179.3M
KURT.	-1.25	-1.25
SKEW	-0.223	-0.135
SUM	314.0M	427.3M

2 age

VALUES:	12,786 (>99%)	17,987 (>99%)
MISSING:	4 (<1%)	10 (<1%)
DISTINCT:	56 (<1%)	74 (<1%)
ZEROES:	---	---

MAX	86.0	95.0
95%	49.0	56.0
Q3	37.0	43.0
AVG	33.7	37.2
MEDIAN	32.0	35.0
Q1	29.0	30.0
5%	24.0	25.0
MIN	18.0	18.0

RANGE	68.0	77.0
IQR	8.00	13.0
STD	7.60	10.0
VAR	57.8	101
KURT.	2.16	1.26
SKEW	1.21	1.05
SUM	431k	669k

3 salary

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	11 (<1%)	11 (<1%)
ZEROES:	68 (<1%)	85 (<1%)

MAX	120k	120k
95%	100k	100k
Q3	70k	70k
MEDIAN	60k	60k
AVG	57k	58k
Q1	20k	20k
5%	4k	4k
MIN	0k	0k

RANGE	120k	120k
IQR	50,000	50,000
STD	32,317	31,675
VAR	1.0B	1.0B
KURT.	-0.991	-0.920
SKEW	-0.042	-0.024
SUM	732.7M	1.0B

4 balance

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	3,771 (29%)	4,490 (25%)
ZEROES:	772 (6%)	1,239 (7%)

MAX	102k	102k
95%	6k	5k
Q3	1k	1k
AVG	1k	1k
MEDIAN	0k	0k
Q1	0k	0k
5%	-0k	-0k
MIN	-8k	-8k

RANGE	110k	110k
IQR	1,246	1,228
STD	2,875	2,891
VAR	8.3M	8.4M
KURT.	178	162
SKEW	8.89	8.84
SUM	16.6M	22.8M

5 marital

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	1 (<1%)	2 (<1%)

div

6 **targeted**

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	2 (<1%)	2 (<1%)

7 **default**

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	2 (<1%)	2 (<1%)

8 **housing**

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	2 (<1%)	2 (<1%)

9 **loan**

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	2 (<1%)	2 (<1%)

10 **contact**

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	3 (<1%)	3 (<1%)

c
unk
telep

11 **day**

VALUES:	12,790 (100%)	17,997 (100%)	MAX	31.0	31.0
MISSING:	---	---	95%	29.0	29.0
			Q3	22.0	22.0

RANGE	30.0	30.0
IQR	14.0	14.0
STD	8.55	8.46

12 month

VALUES:	12,774 (>99%)	17,978 (>99%)
MISSING:	16 (<1%)	19 (<1%)
DISTINCT:	12 (<1%)	12 (<1%)

may,
jul,
aug,
jun,
(

13 duration

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	1,891 (15%)	2,108 (12%)

43	<1%	59	<1%	1.68333333333333 min
42	<1%	52	<1%	1.5 min
41	<1%	60	<1%	1.98333333333333 min
41	<1%	52	<1%	2.63333333333333 min
40	<1%	53	<1%	1.2 min
39	<1%	48	<1%	1.65 min
39	<1%	47	<1%	1.58333333333333 min
12,505	98%	17,626	98%	(Other)

14 campaign

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	42 (<1%)	42 (<1%)
ZEROES:	---	---

MAX	58.0	58.0	RANGE	57.0	57.0
95%	7.0	7.0	IQR	2.00	2.00
Q3	3.0	3.0	STD	3.11	3.02
AVG	2.6	2.6	VAR	9.65	9.12
MEDIAN	2.0	2.0			
Q1	1.0	1.0	KURT.	43.8	40.8
5%	1.0	1.0	SKEW	5.23	5.04
MIN	1.0	1.0	SUM	33,891	47,590

15 pdays

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	441 (3%)	469 (3%)
ZEROES:	---	---

MAX	850	871	RANGE	851	872
95%	334	332	IQR	0.00	0.00
Q3	-1	-1	STD	105	104
AVG	45	44	VAR	11,034	10,873
MEDIAN	-1	-1			
Q1	-1	-1	KURT.	5.37	5.85
5%	-1	-1	SKEW	2.41	2.47
MIN	-1	-1	SUM	571k	785k

16 previous

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	35 (<1%)	36 (<1%)
ZEROES:	10,209 (80%)	14,485 (80%)

MAX	51.0	51.0	RANGE	51.0	51.0
95%	4.0	4.0	IQR	0.00	0.00
Q3	0.0	0.0	STD	2.08	2.01
AVG	0.6	0.6	VAR	4.34	4.04
MEDIAN	0.0	0.0			
Q1	0.0	0.0	KURT.	94.0	89.6
5%	0.0	0.0	SKEW	7.52	7.36
MIN	0.0	0.0	SUM	8,220	11,091

17 poutcome

18 **response**

VALUES:	12,782 (>99%)	17,986 (>99%)
MISSING:	8 (<1%)	11 (<1%)
DISTINCT:	2 (<1%)	2 (<1%)

19 **job**

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	12 (<1%)	12 (<1%)

manage
techn
blue-
a
(

20 **education**

VALUES:	12,790 (100%)	17,997 (100%)
MISSING:	---	---
DISTINCT:	4 (<1%)	4 (<1%)

seco
te
pr
unk