데이터마이닝 과제1

- 이종법 12171676 컴퓨터공학과

포르투칼 소매은행 정기예금 텔레마케팅 결과

다른 변수들이 정기예금 가입 여부에 어떤 영향을 주는지 EDA를 통하여 분석하라.

변수 설명

```
   1 - age : 나이 (numeric)
   2 - job : 직업종류 (categorical)
admin., blue-collar, entrepreneur, housemaid, management, retired, self-employed,
services, student, technician, unemployed, unknown.
   3 - marital : 결혼 상태 (categorical)
divorced, married, single, unknown.
note: divorced means divorced or widowed
   4 - education : 교육 정도 (categorical)
basic.4y, basic.6y, basic.9y, high.school, illiterate, professional.course,
university.degree, unknown.
   5 - default: 신용불량 여부 (categorical)
no, yes, unknown.
   6 - housing: 주택대출 여부 (categorical)
no, yes, unknown.
   7 - loan: 개인대출 여부 (categorical)
no, yes, unknown.

   # related with the last contact of the current campaign:
   8 - contact: 접촉 유형 (categorical)
cellular, telephone
   9 - month: 마직막 접촉 월 (categorical)
jan,  feb,  mar,  ...,  nov,  dec.
  10 - day_of_week: 마지막 접촉 요일 (categorical)
mon, tue, wed, thu, fri.
  11 - duration: 마지막 접촉 통화시간 (단위:초) (numeric)

   # other attributes:
  12 - campaign: 이 캠페인에서 해당 클라이언트에 대한 접촉 횟수 (numeric)
includes last contact
  13 - pdays: 이전 캠페인에서 클라이언트에 마지막으로 연락한 후 경과한 일 수 (numeric)
999 클라이언트에 이전에 연결되지 않음을 의미함
  14 - previous: 지난 캠페인에서 해당 클라이언트에 대한 접촉 횟수 (numeric)
  15 - poutcome: 이전 마케팅 캠페인의 결과 (categorical)
failure, nonexistent, success

   # social and economic context attributes
  16 - emp.var.rate: 고용 변동률 - 분기별 지표 (numeric)
  17 - cons.price.idx: 소비자물가지수 - 월간지표 (numeric)
  18 - cons.conf.idx: 소비자 신뢰 지수 - 월간 지표 (numeric)
  19 - euribor3m: 3개월 만기 유로예금 이자율 - 일별 지표 (numeric)
  20 - nr.employed: 고용자 수 - 분기별 지표 (numeric)

  Target
```

```
  21 - y - 정기예금 가입 여부 (binary)
yes, no
```

category: 10, numeric : 10

- 이를 통해 생각해본 가설 및 변수 해석들
  - age에 따라 그룹이 구별될 것 같다. 가중치가 있는 그룹에 더미변수를 추가할것. (변수 가공)
  - (month, day_of_week)는 현재 조사시점이 언제인지와 관련이 있어보인다.pdays와 다중공선성이 있을것이라 생각됨. 분포 확인 후 제거. (변수가공)
  - unknown이 있는 컬럼들은 one-hot-encoding이 되야한다.
  - pdays는 실제 통화후 시간, previous는 실제 통화를 안해도 연락한 횟수를 말한다.
  - 16~20는 주변 상황을 의미하는 변수로, 선형 변환으로 하나로 그룹 되지 않을까? (변수 가공)
  - default 신용불량 여부 unknown은, 신용불량자가 답하지 않을 것으로 생각. 정기예금을 가입하지 않을 확률이 높다 생각한다. unknonwn은 yes로 바꿔줄 수 있지 않을까?

In [2]:
```
import matplotlib.pyplot as plt
import numpy as np
from glob import glob
import pandas as pd
import sys
```

In [3]:
```
#시각화 패키지들
import seaborn as sns
import matplotlib.pyplot as plt
from statsmodels.graphics.mosaicplot import mosaic
%matplotlib inline

plt.style.use("ggplot")
```

In [4]:
```
#분석 관련
from scipy.stats import chi2_contingency
from scipy.stats import kstest
from statsmodels.formula.api import ols
```

데이터 확인

In [5]:
```
# 파일 불러오기
train = pd.read_csv('./data/bank-full2.csv')
#submission = pd.read_csv('./data/sample_submission.csv')
```

더 자세하게 확인하려면, train과 test를 나눠야겠지만 과제이기 때문에 train으로 분석

In [6]:
```
train
```

Out[6]:

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	...	c
0	56	housemaid	married	basic.4y	no	no	no	telephone	may	mon	...	
1	57	services	married	high.school	unknown	no	no	telephone	may	mon	...	
2	37	services	married	high.school	no	yes	no	telephone	may	mon	...	
3	40	admin.	married	basic.6y	no	no	no	telephone	may	mon	...	

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	...	c
4	56	services	married	high.school	no	no	yes	telephone	may	mon	...	
...	...	...	...	...	...	...	...	...	...	...	...	
41183	73	retired	married	professional.course	no	yes	no	cellular	nov	fri	...	
41184	46	blue-collar	married	professional.course	no	no	no	cellular	nov	fri	...	
41185	56	retired	married	university.degree	no	yes	no	cellular	nov	fri	...	
41186	44	technician	married	professional.course	no	no	no	cellular	nov	fri	...	
41187	74	retired	married	professional.course	no	yes	no	cellular	nov	fri	...	

41188 rows × 21 columns

pdays가 999인데, prev

In [10]:
```
#차원 확인
print("all data: ",train.shape)
```

```
all data:  (41188, 21)
```

In [15]:
```
print("yes : ",sum(train['y']=='yes'))
print("no : ",sum(train['y']=='no'))
```

```
yes :  4640
no :  36548
```

라벨에 불균형이 있음을 볼 수 있다.

데이터 타입 확인

In [6]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41188 entries, 0 to 41187
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   age             41188 non-null  int64  
 1   job             41188 non-null  object 
 2   marital         41188 non-null  object 
 3   education       41188 non-null  object 
 4   default         41188 non-null  object 
 5   housing         41188 non-null  object 
 6   loan            41188 non-null  object 
 7   contact         41188 non-null  object 
 8   month           41188 non-null  object 
 9   day_of_week     41188 non-null  object 
 10  duration        41188 non-null  int64  
 11  campaign        41188 non-null  int64  
 12  pdays           41188 non-null  int64  
 13  previous        41188 non-null  int64  
 14  poutcome        41188 non-null  object 
 15  emp.var.rate    41188 non-null  float64
 16  cons.price.idx  41188 non-null  float64
 17  cons.conf.idx   41188 non-null  float64
 18  euribor3m       41188 non-null  float64
 19  nr.employed     41188 non-null  float64
 20  y               41188 non-null  object 
```

```
dtypes: float64(5), int64(5), object(11)
memory usage: 6.6+ MB
```

- na는 없어 보인다. 하지만 내부에 unknown이 존재.

In [7]:
```
#컬럼명 변경
# . 이 있으면 분석시 오류가 생겨서.
train.columns = train.columns.str.replace('.','_')
```

```
C:\Users\dlwhd\miniconda3\envs\localtorch\lib\site-packages\ipykernel_launcher.py:3: FutureWarning: The default value of regex will change from True to False in a future version. In addition, single character regular expressions will *not* be treated as literal strings when regex=True.
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [19]:
```
train.isna().sum()
```

Out[19]:
```
age               0
job               0
marital           0
education         0
default           0
housing           0
loan              0
contact           0
month             0
day_of_week       0
duration          0
campaign          0
pdays             0
previous          0
poutcome          0
emp_var_rate      0
cons_price_idx    0
cons_conf_idx     0
euribor3m         0
nr_employed       0
y                 0
dtype: int64
```

In [22]:
```
(train=='unknown').sum()
```

Out[22]:
```
age                 0
job               330
marital            80
education        1731
default          8597
housing           990
loan              990
contact             0
month               0
day_of_week         0
duration            0
campaign            0
pdays               0
previous            0
poutcome            0
emp_var_rate        0
cons_price_idx      0
cons_conf_idx       0
euribor3m           0
nr_employed         0
```

```
y                    0
dtype: int64
```

na가 많은 편으로 볼 수 있다.

특이하게, housing과 loan이 같은수의 na를 가진다. 이는, 주택대출 한사람이 개인대출도 했을 가능성이 높다고 보인다. 둘다 unknown을 yes로 바꿔줄 수 있지 않을까.

default는 신용불량자를 뜻하는데 target에 따른 분포를 확인해서, yes로 바꿔줄 수 있다고 생각함.

범주형 레벨 확인

In [23]:
```
print(train.columns)
```

```
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 [24]:
```
print('job \n',train['job'].unique())
print('marital \n',train['marital'].unique())
print('education \n',train['education'].unique())
print('default \n',train['default'].unique())
print('housing \n',train['housing'].unique())
print('loan \n',train['loan'].unique())
print('contact \n',train['contact'].unique())
print('month \n',train['month'].unique())
print('day_of_week \n',train['day_of_week'].unique())
print('poutcome \n',train['poutcome'].unique())
```

```
job
 ['housemaid' 'services' 'admin.' 'blue-collar' 'technician' 'retired'
 'management' 'unemployed' 'self-employed' 'unknown' 'entrepreneur'
 'student']
marital
 ['married' 'single' 'divorced' 'unknown']
education
 ['basic.4y' 'high.school' 'basic.6y' 'basic.9y' 'professional.course'
 'unknown' 'university.degree' 'illiterate']
default
 ['no' 'unknown' 'yes']
housing
 ['no' 'yes' 'unknown']
loan
 ['no' 'yes' 'unknown']
contact
 ['telephone' 'cellular']
month
 ['may' 'jun' 'jul' 'aug' 'oct' 'nov' 'dec' 'mar' 'apr' 'sep']
day_of_week
 ['mon' 'tue' 'wed' 'thu' 'fri']
poutcome
 ['nonexistent' 'failure' 'success']
```

오타가 있는 범주는 없다.

범주형 데이터 분석

job

- unknow값도 존재해서, group에 포함시킬 수도 있는지 확인해본다.

target vs job

In [36]:
```
job_crosstab=pd.crosstab(train['job'],train['y'])
job_crossgtab_res=chi2_contingency(job_crosstab)
```

In [37]:
```
job_crosstab
```

Out[37]:

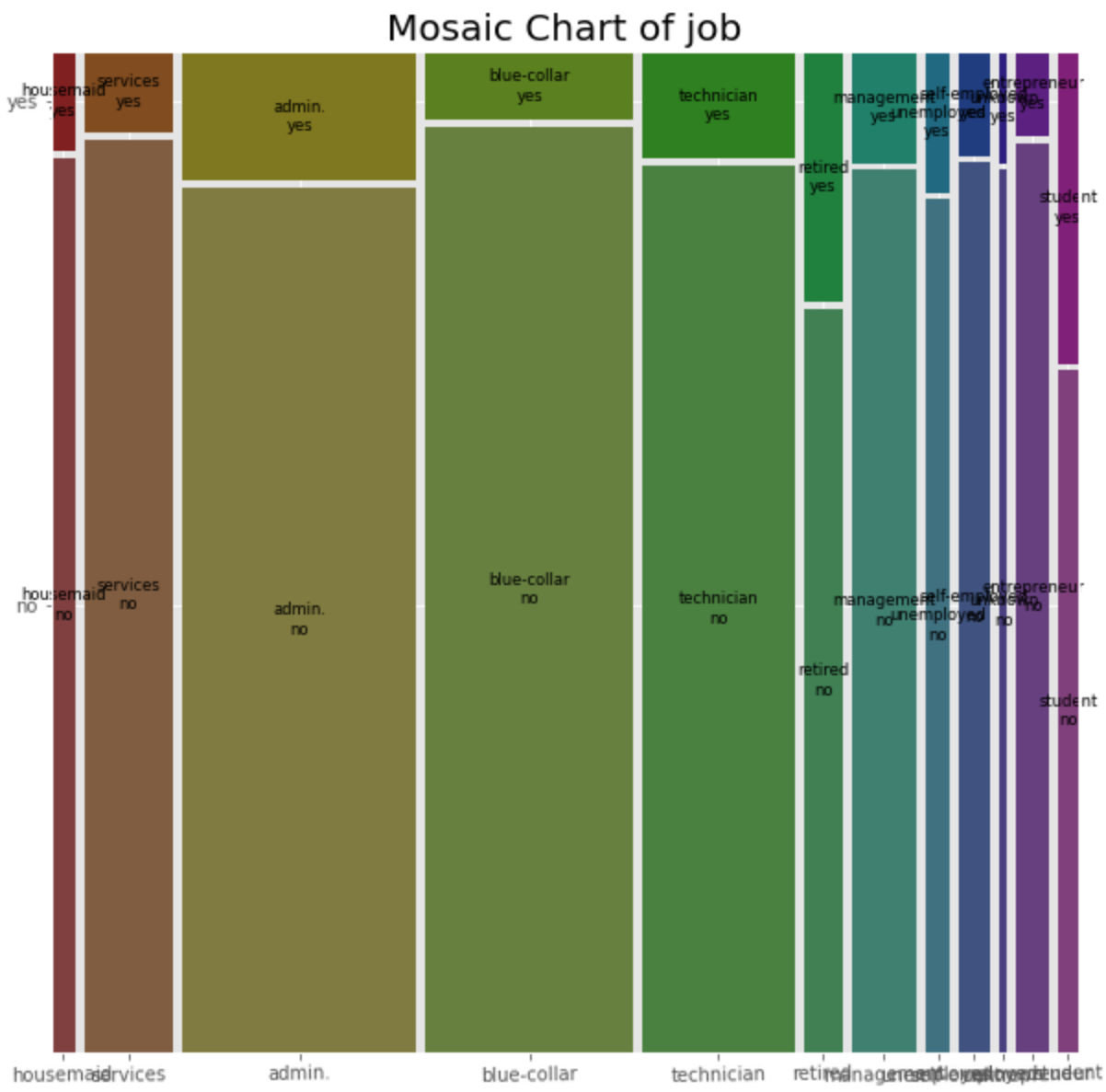
y	no	yes
job		
admin.	9070	1352
blue-collar	8616	638
entrepreneur	1332	124
housemaid	954	106
management	2596	328
retired	1286	434
self-employed	1272	149
services	3646	323
student	600	275
technician	6013	730
unemployed	870	144
unknown	293	37

In [38]:
```
plt.rcParams["figure.figsize"]=(10,10)

mosaic(train, ['job', 'y'],gap=0.01,label_rotation=True)


plt.title('Mosaic Chart of job', fontsize=20)

plt.show()
plt.rcParams["figure.figsize"]=(6.0,4.0)
```



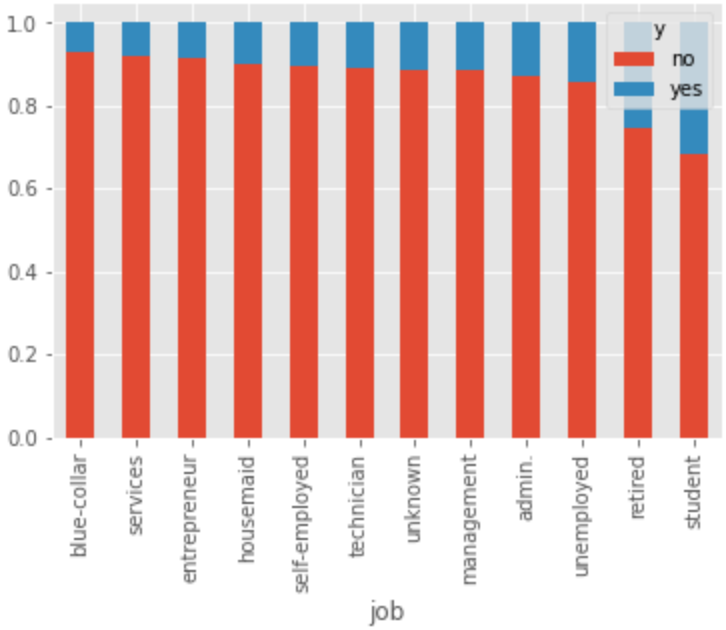
도수만 다를 뿐, student, retired을 제외하고는 다들 비슷한 분포를 가진다.

In [50]:
```
#ratio plot 제작
def ratio_table(cross_tab):
    return cross_tab.div(cross_tab.sum(axis=1),axis='index')

job_ratio=ratio_table(job_crosstab)
job_ratio=job_ratio.sort_values(by='no',ascending=False)

job_ratio.plot(kind='bar',stacked=True)
```

Out[50]:
```
<AxesSubplot:xlabel='job'>
```



In [51]:
```
job_ratio
```

Out[51]:

job	y no	yes
blue-collar	0.931057	0.068943
services	0.918619	0.081381
entrepreneur	0.914835	0.085165
housemaid	0.900000	0.100000
self-employed	0.895144	0.104856
technician	0.891740	0.108260
unknown	0.887879	0.112121
management	0.887825	0.112175
admin.	0.870274	0.129726
unemployed	0.857988	0.142012
retired	0.747674	0.252326
student	0.685714	0.314286

In [28]:
```
print('Chi2 Statistic: {}, p-value: {}'.format(job_crossgtab_res[0], job_crossgtab_res[1])
print('기각')
```

```
Chi2 Statistic: 961.2424403289555, p-value: 4.189763287563623e-199
기각
```

검정 결과 job과 target은 독립이 아님을 알 수 있다.

또한, management와 unknown이 상당히 비슷해 보임을 알 수 있다.

(retired) vs (student) vs (그 외)로 grouping을 진행

In [53]:
```
train_jobs = train.copy()
train_jobs['job']=train_jobs['job'].replace(['blue-collar','services','entrepreneur','hous
```

```
                                'unknown','management','admin.','unemployed'],'job1')
train_jobs['job'].unique()
```

Out[53]:
```
array(['job1', 'retired', 'student'], dtype=object)
```

In [56]:
```
job_crosstab=pd.crosstab(train_jobs['job'],train['y'])
job_crossgtab_res=chi2_contingency(job_crosstab)
```

In [57]:
```
print('Chi2 Statistic: {}, p-value: {}'.format(job_crossgtab_res[0], job_crossgtab_res[1]))
print('기각')
```

```
Chi2 Statistic: 736.5282532641304, p-value: 1.1612398226593737e-160
기각
```

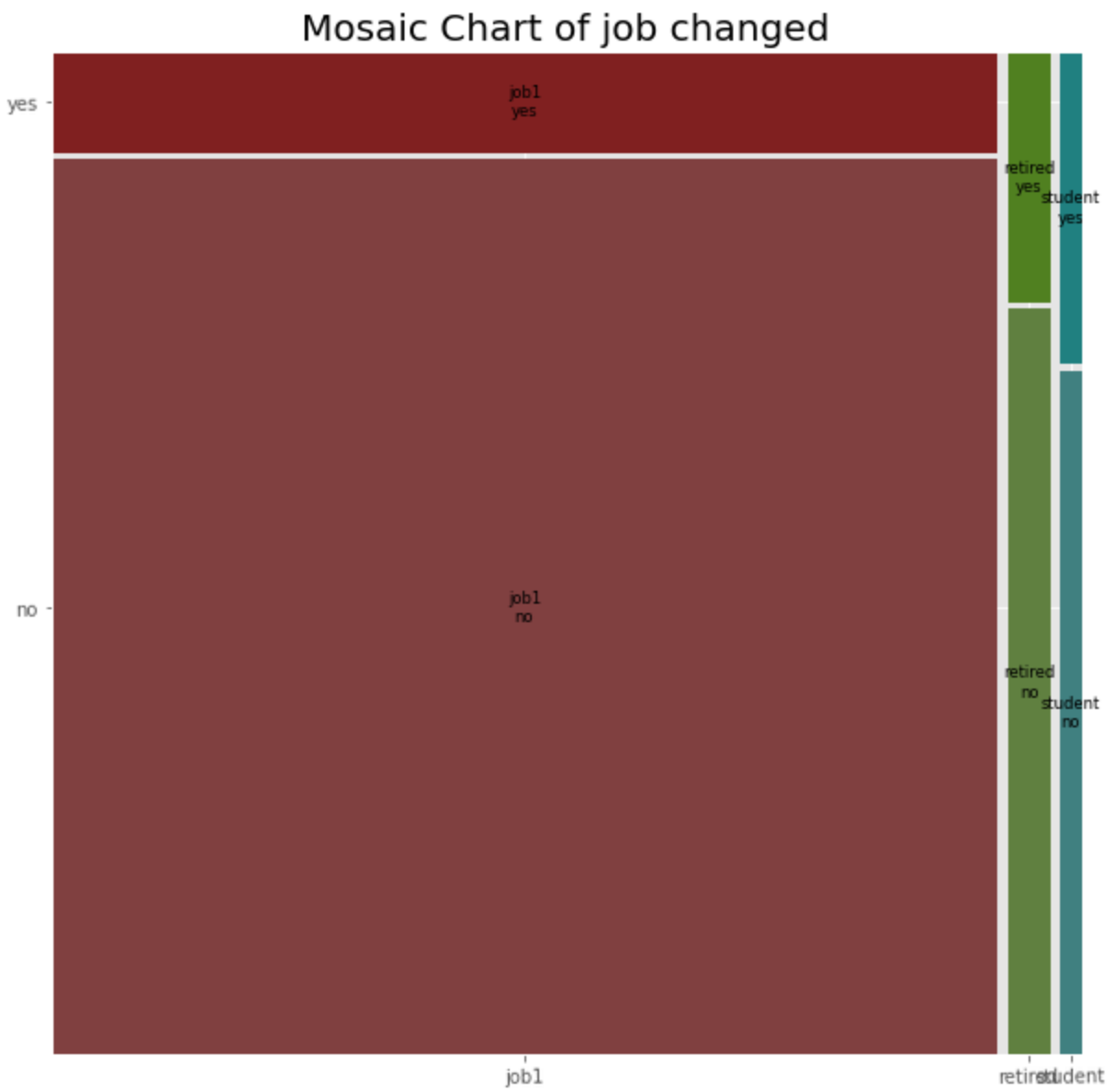
기각하긴 하지만, pvalue가 증가하게 되었다.

In [60]:
```
plt.rcParams["figure.figsize"]=(10,10)

mosaic(train_jobs, ['job', 'y'],gap=0.01,label_rotation=True)


plt.title('Mosaic Chart of job changed', fontsize=20)

plt.show()
plt.rcParams["figure.figsize"]=(6.0,4.0)
```



marital

In [67]:
```
marital_crosstab=pd.crosstab(train['marital'],train['y'])
marital_crosstab_res=chi2_contingency(marital_crosstab)
```

In [68]:
```
marital_crosstab
```

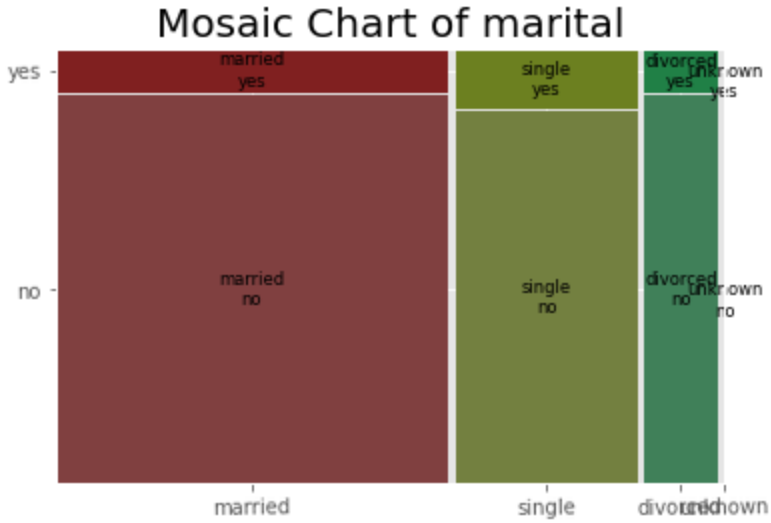
Out[68]:

y	no	yes
marital		
divorced	4136	476
married	22396	2532
single	9948	1620
unknown	68	12

In [69]:
```
mosaic(train, ['marital', 'y'],gap=0.01,label_rotation=True)


plt.title('Mosaic Chart of marital', fontsize=20)
```

```
plt.show()
```



육안으로는 결혼 여부는 큰 영향을 안주는 것으로 관측된다.

In [71]:
```
marital_crosstab
```

Out[71]:

y	no	yes
marital		
divorced	4136	476
married	22396	2532
single	9948	1620
unknown	68	12

In [72]:
```
print('Chi2 Statistic: {}, p-value: {}'.format(marital_crosstab_res[0], marital_crosstab_r
print('기각')
```

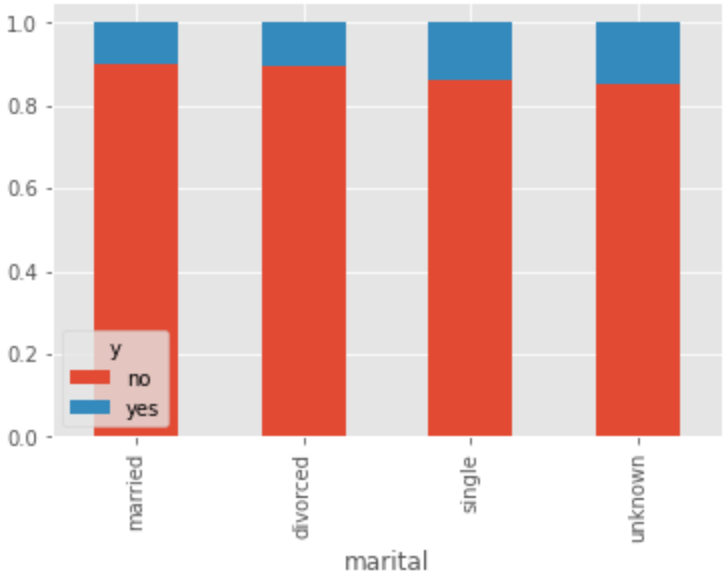
```
Chi2 Statistic: 122.65515182252989, p-value: 2.068014648442211e-26
기각
```

그래도 카이제곱 검정은 기각을 한다. 결혼여부에 따라 다르다고 할 수 있다.

In [73]:
```
marital_ratio=ratio_table(marital_crosstab)
marital_ratio=marital_ratio.sort_values(by='no',ascending=False)

marital_ratio.plot(kind='bar',stacked=True)
```

Out[73]:
```
<AxesSubplot:xlabel='marital'>
```



In [74]:
```
marital_ratio
```

Out[74]:

y	no	yes
marital		
married	0.898427	0.101573
divorced	0.896791	0.103209
single	0.859959	0.140041
unknown	0.850000	0.150000

married와 divorced는 거의 같다고 할 수 있어서, 둘을 합쳐준다.

unknown과 single도 매우 유사하다.

In [79]:
```
train_marital = train.copy()
train_marital['marital']=train_marital['marital'].replace(['married','divorced'],'marital1
train_marital['marital']=train_marital['marital'].replace(['single','unknown'],'marital2')
train_marital['marital'].unique()
```

Out[79]:
```
array(['marital1', 'marital2'], dtype=object)
```

In [80]:
```
marital_crosstab=pd.crosstab(train_marital['marital'],train['y'])
marital_crosstab_res=chi2_contingency(marital_crosstab)
print('Chi2 Statistic: {}, p-value: {}'.format(marital_crosstab_res[0], marital_crosstab_r
print('기각')
```

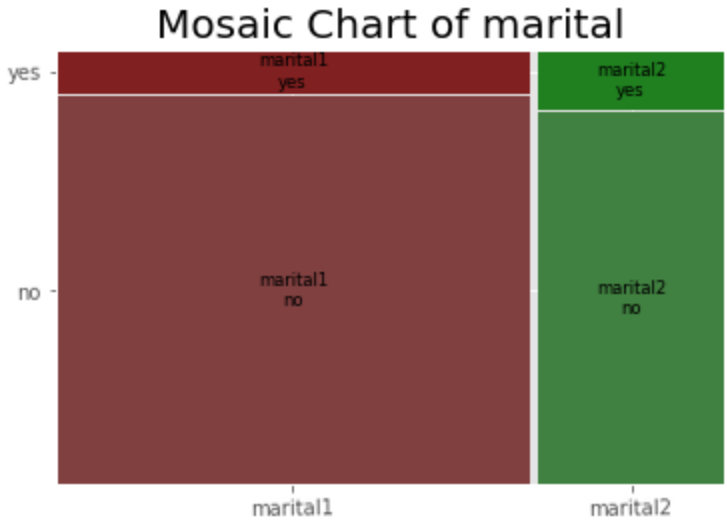
```
Chi2 Statistic: 122.08940156886162, p-value: 2.206669293572927e-28
기각
```

grouping 후에 오히려, p-value가 감소했다.

In [81]:
```
mosaic(train_marital, ['marital', 'y'],gap=0.01,label_rotation=True)


plt.title('Mosaic Chart of marital', fontsize=20)

plt.show()
```



education

In [107…
```
education_crosstab=pd.crosstab(train['education'],train['y'])
education_crosstab_res=chi2_contingency(education_crosstab)
```

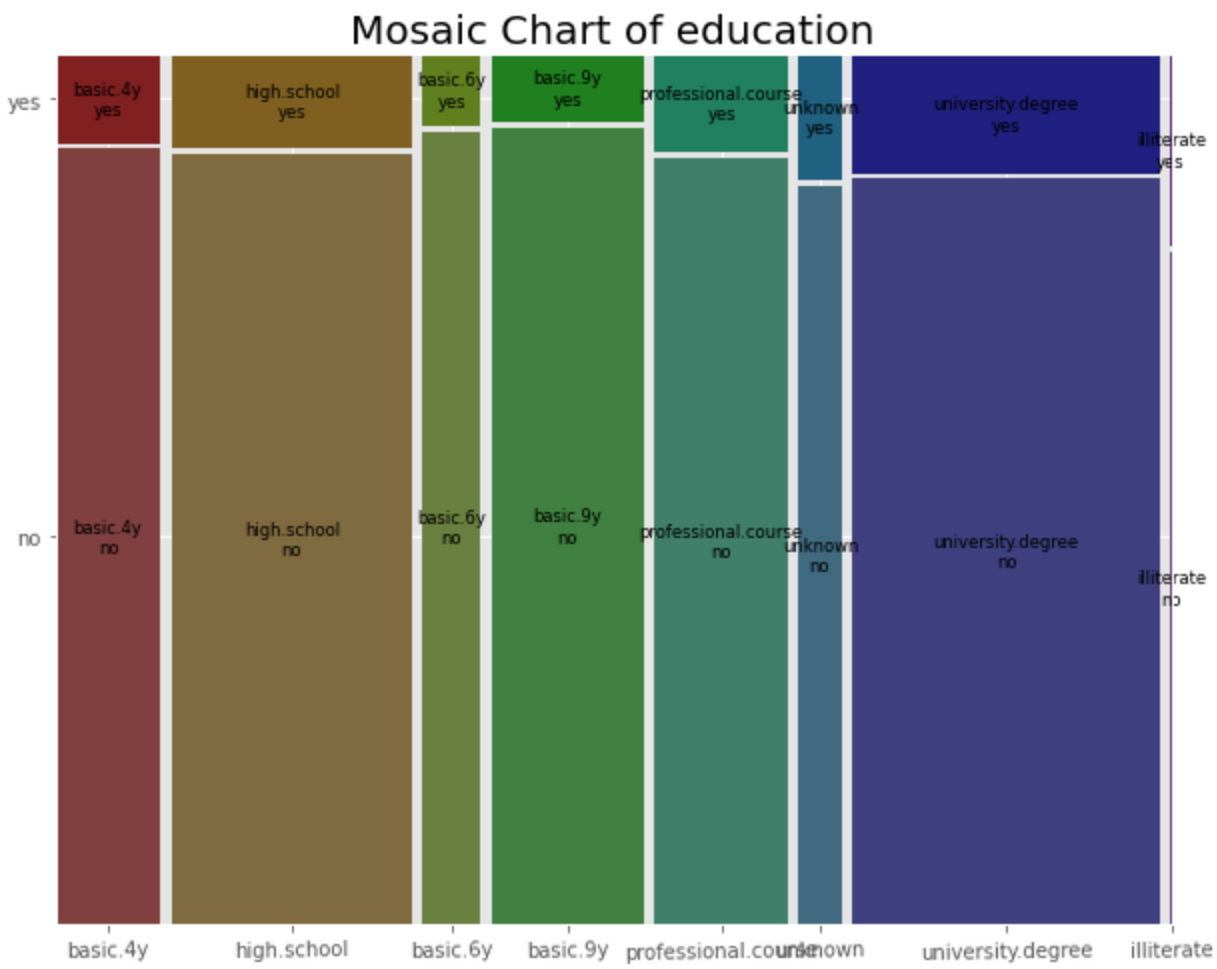
In [108…
```
education_crosstab
```

Out[108…

y	no	yes
education		
basic.4y	3748	428
basic.6y	2104	188
basic.9y	5572	473
high.school	8484	1031
illiterate	14	4
professional.course	4648	595
university.degree	10498	1670
unknown	1480	251

In [109…
```
plt.rcParams["figure.figsize"]=(10,8)
mosaic(train, ['education', 'y'],gap=0.01,label_rotation=True)
plt.title('Mosaic Chart of education', fontsize=20)

plt.show()
plt.rcParams["figure.figsize"]=(6.0,4.0)
```



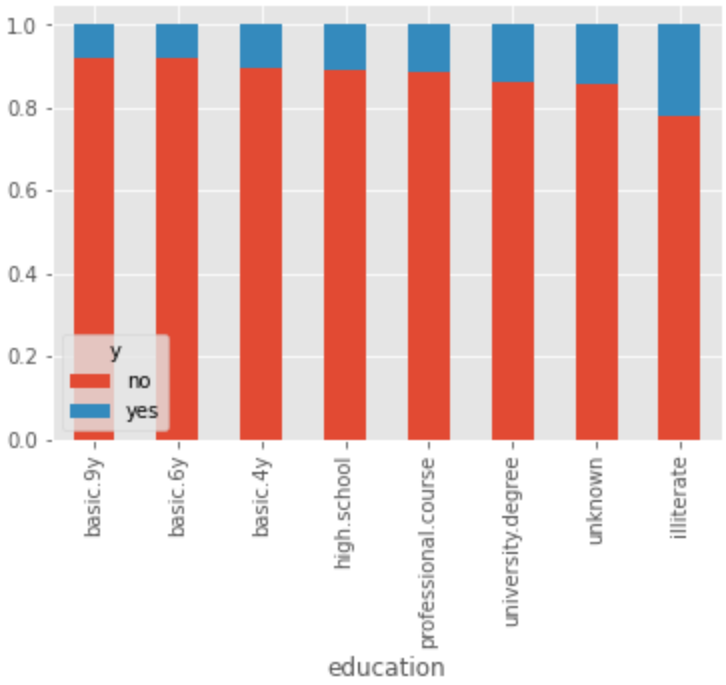
In [110…

```
#ratio plot 제작
education_ratio=ratio_table(education_crosstab)
education_ratio=education_ratio.sort_values(by='no',ascending=False)

education_ratio.plot(kind='bar',stacked=True)
```

Out[110…

```
<AxesSubplot:xlabel='education'>
```



In [111…

```
print('Chi2 Statistic: {}, p-value: {}'.format(education_crosstab_res[0], education_crosst
```

```
print('기각')
```

```
Chi2 Statistic: 193.10590454149565, p-value: 3.3051890144025054e-38
기각
```

In [112…
```
education_ratio
```

Out[112…

y	no	yes
education		
basic.9y	0.921754	0.078246
basic.6y	0.917976	0.082024
basic.4y	0.897510	0.102490
high.school	0.891645	0.108355
professional.course	0.886515	0.113485
university.degree	0.862755	0.137245
unknown	0.854997	0.145003
illiterate	0.777778	0.222222

basic과 highschool / profess / university, unknown / illiterate로 나눠보자

In [113…
```
train_education = train.copy()
train_education['education']=train_education['education'].replace(['basic.9y','basic.6y',
train_education['education']=train_education['education'].replace(['university.degree','ur
train_education['education'].unique()
```

Out[113…
```
array(['education1', 'professional.course', 'university.degree',
       'illiterate'], dtype=object)
```

In [114…
```
education_crosstab=pd.crosstab(train_education['education'],train['y'])
education_crosstab_res=chi2_contingency(education_crosstab)
```

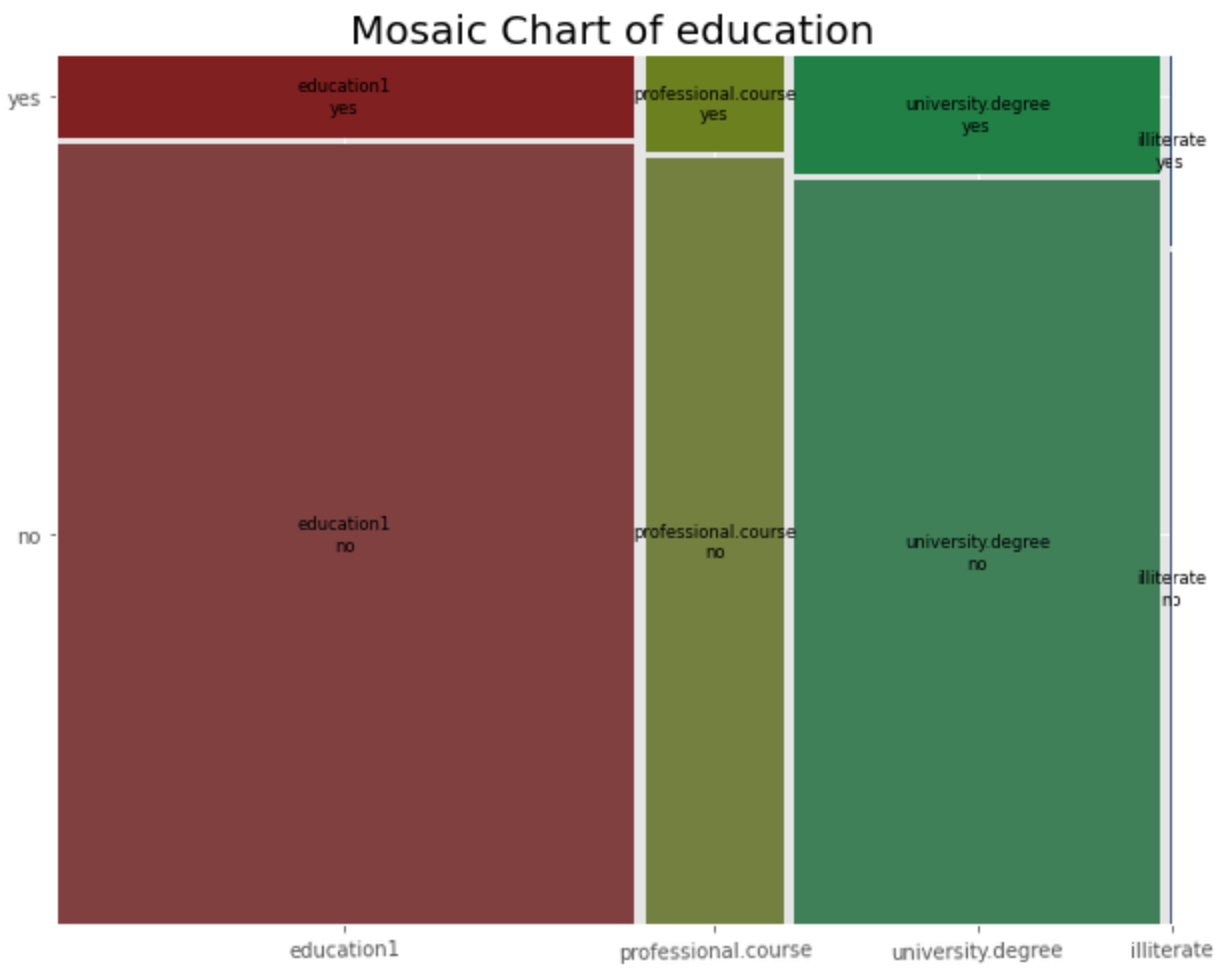
In [115…
```
education_crosstab
```

Out[115…

y	no	yes
education		
education1	19908	2120
illiterate	14	4
professional.course	4648	595
university.degree	11978	1921

In [116…
```
plt.rcParams["figure.figsize"]=(10,8)
mosaic(train_education, ['education', 'y'],gap=0.01,label_rotation=True)
plt.title('Mosaic Chart of education', fontsize=20)

plt.show()
plt.rcParams["figure.figsize"]=(6.0,4.0)
```



In [106…
```
print('Chi2 Statistic: {}, p-value: {}'.format(education_crosstab_res[0], education_crosst
print('기각')
```

```
Chi2 Statistic: 152.37803568673755, p-value: 8.086373337274164e-33
기각
```

잘 구분되었음을 볼 수 있다.

default

가설로 unknown은 신용불량자가 아닐까 제안했다. 그것을 확인해본다.

In [117…
```
default_crosstab=pd.crosstab(train['default'],train['y'])
default_crosstab_res=chi2_contingency(default_crosstab)
```

In [119…
```
default_crosstab
```

Out[119…

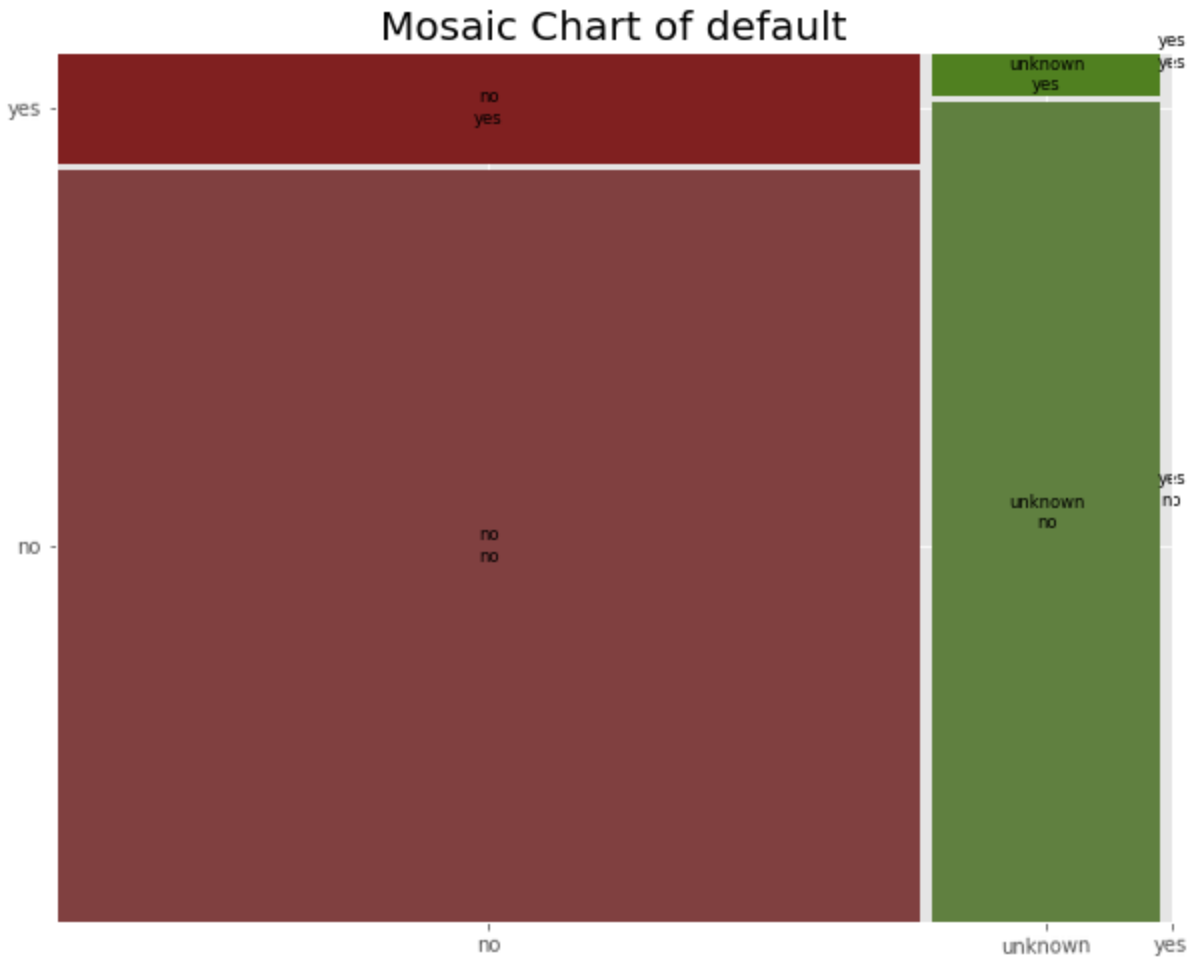
y	no	yes
default		
no	28391	4197
unknown	8154	443
yes	3	0

In [122…
```
plt.rcParams["figure.figsize"]=(10,8)
```

```
mosaic(train, ['default', 'y'],gap=0.01,label_rotation=True)


plt.title('Mosaic Chart of default', fontsize=20)

plt.show()
plt.rcParams["figure.figsize"]=(6,4)
```



스스로 신용불량자라 말하지 못한 것으로 보이고, 다들 unknown에 있다.

In [123…
```
train_default = train.copy()
train_default['default']=train_default['default'].replace(['unknown'],'yes')
train_default['default'].unique()
```

Out[123…
```
array(['no', 'yes'], dtype=object)
```

housing & loan

가설로 housing과 loan이 둘다 unknown이면 yes일 수 있다고 생각.

In [125…
```
housing_crosstab=pd.crosstab(train['housing'],train['y'])
housing_crosstab_res=chi2_contingency(housing_crosstab)

loan_crosstab=pd.crosstab(train['loan'],train['y'])
loan_crosstab_res=chi2_contingency(loan_crosstab)
```

In [126…
```
housing_crosstab
```

Out[126…

y	no	yes
housing		
no	16596	2026
unknown	883	107
yes	19069	2507

In [128…

```
loan_crosstab
```

Out[128…

y	no	yes
loan		
no	30100	3850
unknown	883	107
yes	5565	683

In [129…

```
print('housing Chi2 Statistic: {}, p-value: {}'.format(housing_crosstab_res[0], housing_cr
print('loan Chi2 Statistic: {}, p-value: {}'.format(loan_crosstab_res[0], loan_crosstab_re
```

```
housing Chi2 Statistic: 5.684495858974168, p-value: 0.05829447669453452
loan Chi2 Statistic: 1.094027551150338, p-value: 0.5786752870441754
```

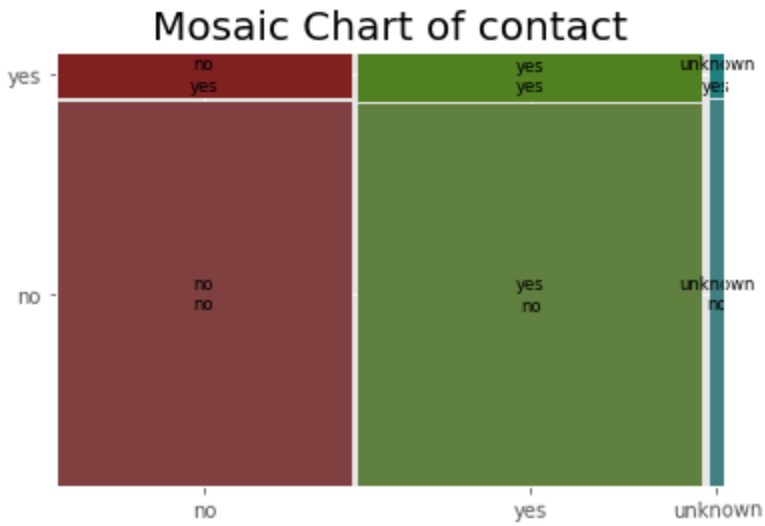
!중요 loan과 housing은 구분력이 없음을 볼 수있다.

In [135…

```
mosaic(train, ['housing', 'y'],gap=0.01,label_rotation=True)
plt.title('Mosaic Chart of contact', fontsize=20)
```

Out[135…

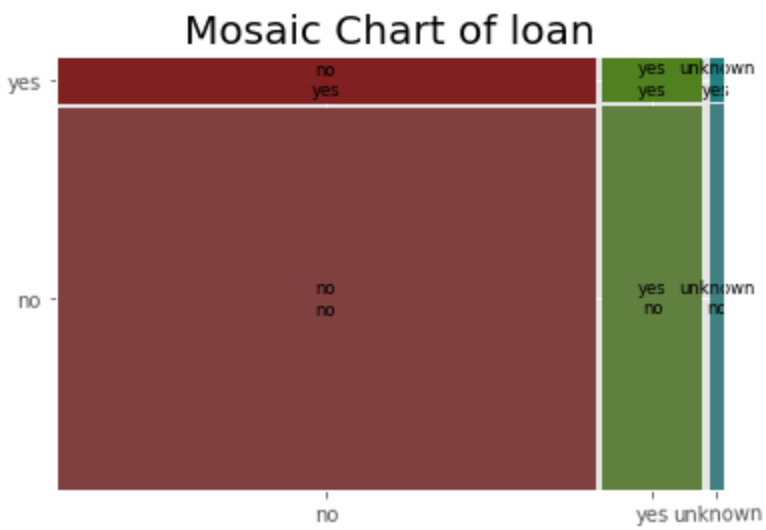
```
Text(0.5, 1.0, 'Mosaic Chart of contact')
```



In [132…

```
mosaic(train, ['loan', 'y'],gap=0.01,label_rotation=True)
plt.title('Mosaic Chart of loan', fontsize=20)

plt.show()
```



housing과 loan은 제거.

last contact

위 3개는 필요없는 컬럼들이라 생각을 했고, 확인해본다.

poutcome은 영향이 크다 생각했다.

- contact ('telephone' 'cellular')
- month ('may' 'jun' 'jul' 'aug' 'oct' 'nov' 'dec' 'mar' 'apr' 'sep')
- day_of_week ('mon' 'tue' 'wed' 'thu' 'fri')
- poutcome ('nonexistent' 'failure' 'success')

In [140…

```
contact_crosstab=pd.crosstab(train['contact'],train['y'])
contact_crosstab_res=chi2_contingency(contact_crosstab)

month_crosstab=pd.crosstab(train['month'],train['y'])
month_crosstab_res=chi2_contingency(month_crosstab)

day_of_week_crosstab=pd.crosstab(train['day_of_week'],train['y'])
day_of_week_crosstab_res=chi2_contingency(day_of_week_crosstab)

poutcome_crosstab=pd.crosstab(train['poutcome'],train['y'])
poutcome_crosstab_res=chi2_contingency(poutcome_crosstab)

print('contact_crosstab Chi2 Statistic: {}, p-value: {}'.format(contact_crosstab_res[0], c
print('month_crosstab Chi2 Statistic: {}, p-value: {}'.format(month_crosstab_res[0], montl
print('day_of_week_crosstab Chi2 Statistic: {}, p-value: {}'.format(day_of_week_crosstab_r
print('poutcome_crosstab_res Chi2 Statistic: {}, p-value: {}'.format(poutcome_crosstab_res
```

```
contact_crosstab Chi2 Statistic: 862.3183642075705, p-value: 1.5259856523129964e-189
month_crosstab Chi2 Statistic: 3101.149351411678, p-value: 0.0
day_of_week_crosstab Chi2 Statistic: 26.14493907587197, p-value: 2.9584820052785324e-05
poutcome_crosstab_res Chi2 Statistic: 4230.5237978319765, p-value: 0.0
```

예상과 다르게 모두 영향이 있다고 나온다.

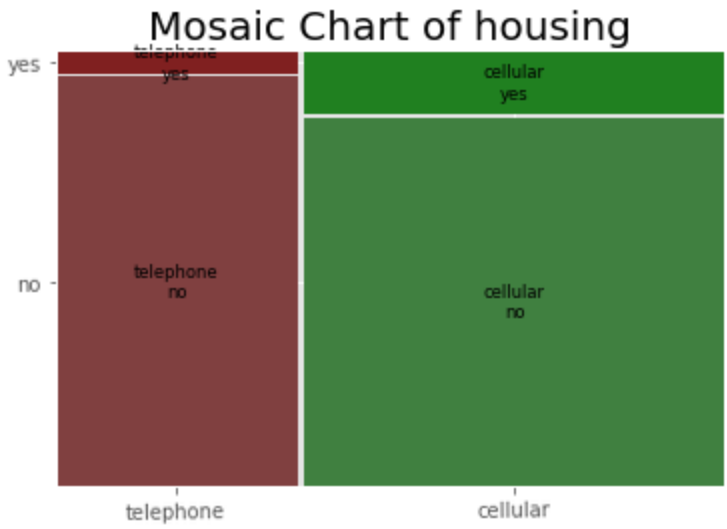
그중, 몇월이냐와 이전 결과가 가장 영향이 크다.

In [136…

```
mosaic(train, ['contact', 'y'],gap=0.01,label_rotation=True)
plt.title('Mosaic Chart of housing', fontsize=20)
```

Out[136…

```
Text(0.5, 1.0, 'Mosaic Chart of housing')
```

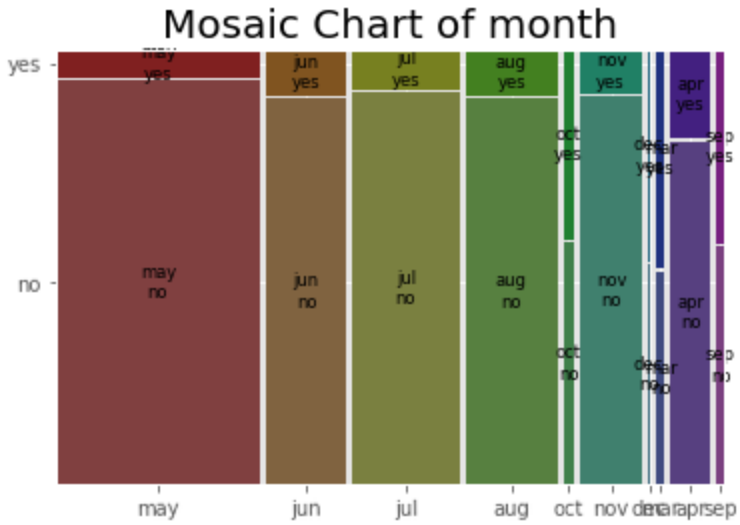


In [137…

```
mosaic(train, ['month', 'y'],gap=0.01,label_rotation=True)
plt.title('Mosaic Chart of month', fontsize=20)
```

Out[137…

```
Text(0.5, 1.0, 'Mosaic Chart of month')
```



월마다, 도수와 비율이 다르다.

In [148…

```
month_crosstab
```

Out[148…

y	no	yes
month		
apr	2093	539
aug	5523	655
dec	93	89
jul	6525	649
jun	4759	559
mar	270	276
may	12883	886
nov	3685	416
oct	403	315

y	no	yes
month		
sep	314	256

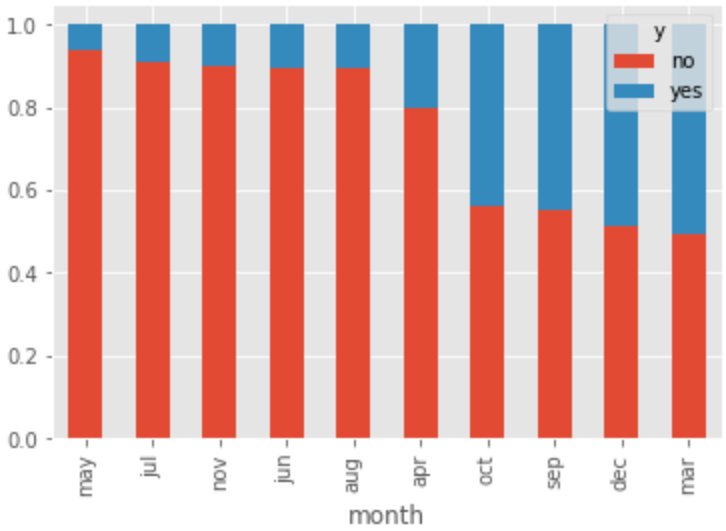
In [145…

```
#ratio plot 제작
month_ratio=ratio_table(month_crosstab)
month_ratio=month_ratio.sort_values(by='no',ascending=False)

month_ratio.plot(kind='bar',stacked=True)
```

Out[145…

```
<AxesSubplot:xlabel='month'>
```



In [149…

```
month_ratio
```

Out[149…

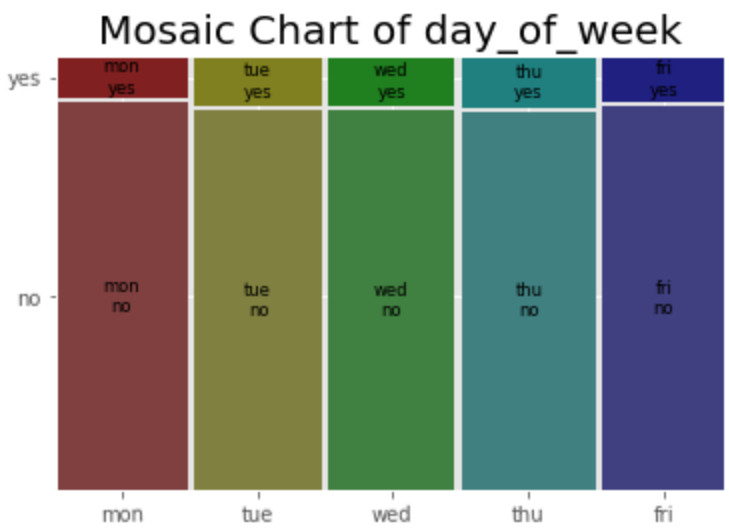
y	no	yes
month		
may	0.935653	0.064347
jul	0.909534	0.090466
nov	0.898561	0.101439
jun	0.894885	0.105115
aug	0.893979	0.106021
apr	0.795213	0.204787
oct	0.561281	0.438719
sep	0.550877	0.449123
dec	0.510989	0.489011
mar	0.494505	0.505495

may / jul,nov,jun, aug / apr / oct,sep / dec,mar/ 로 구분할 수 있다.

In [138…

```
mosaic(train, ['day_of_week', 'y'],gap=0.01,label_rotation=True)
plt.title('Mosaic Chart of day_of_week', fontsize=20)
```

Out[138… Text(0.5, 1.0, 'Mosaic Chart of day_of_week')

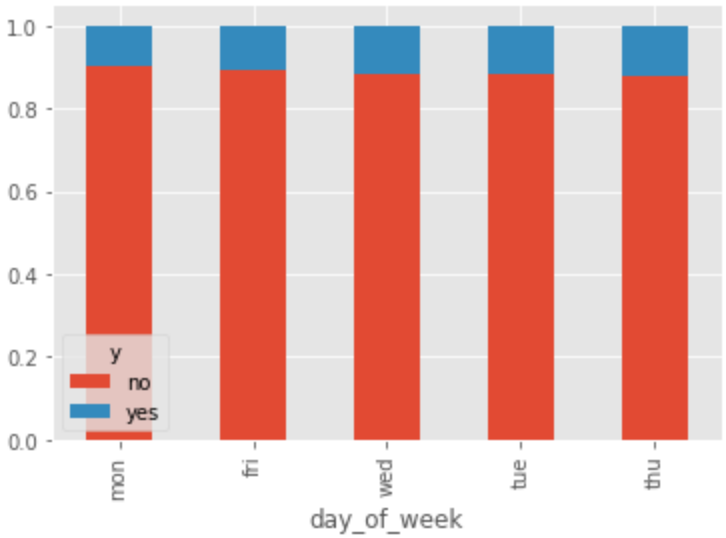


In [141…

```
#ratio plot 제작
day_of_week_ratio=ratio_table(day_of_week_crosstab)
day_of_week_ratio=day_of_week_ratio.sort_values(by='no',ascending=False)

day_of_week_ratio.plot(kind='bar',stacked=True)
```

Out[141… <AxesSubplot:xlabel='day_of_week'>



In [143…

```
day_of_week_crosstab
```

Out[143…

y	no	yes
day_of_week		
fri	6981	846
mon	7667	847
thu	7578	1045
tue	7137	953
wed	7185	949

In [142…

```
day_of_week_ratio
```

Out[142…

y	no	yes
day_of_week		
mon	0.900517	0.099483
fri	0.891913	0.108087
wed	0.883329	0.116671
tue	0.882200	0.117800
thu	0.878812	0.121188

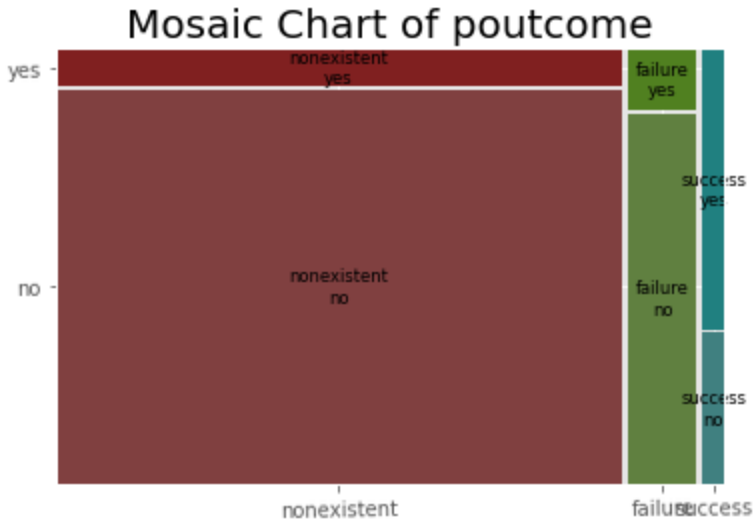
mon,fri / tue,wed,thu 으로 구분이된다.

In [139…

```
mosaic(train, ['poutcome', 'y'],gap=0.01,label_rotation=True)
plt.title('Mosaic Chart of poutcome', fontsize=20)
```

Out[139…

```
Text(0.5, 1.0, 'Mosaic Chart of poutcome')
```



poutcome은 nonexistent 자체가 새로운 값이다.

수치형 데이터 분석

```
  11 - duration: 마지막 접촉 통화시간 (단위:초) (numeric)

   # other attributes:
  12 - campaign: 이 캠페인에서 해당 클라이언트에 대한 접촉 횟수 (numeric)
includes last contact
  13 - pdays: 이전 캠페인에서 클라이언트에 마지막으로 연락한 후 경과한 일 수 (numeric)
999 클라이언트에 이전에 연결되지 않음을 의미함
  14 - previous: 지난 캠페인에서 해당 클라이언트에 대한 접촉 횟수 (numeric)

   # social and economic context attributes
  16 - emp.var.rate: 고용 변동률 - 분기별 지표 (numeric)
  17 - cons.price.idx: 소비자물가지수 - 월간지표 (numeric)
  18 - cons.conf.idx: 소비자 신뢰 지수 - 월간 지표 (numeric)
  19 - euribor3m: 3개월 만기 유로예금 이자율 - 일별 지표 (numeric)
  20 - nr.employed: 고용자 수 - 분기별 지표 (numeric)
```

전체 상관계수 확인

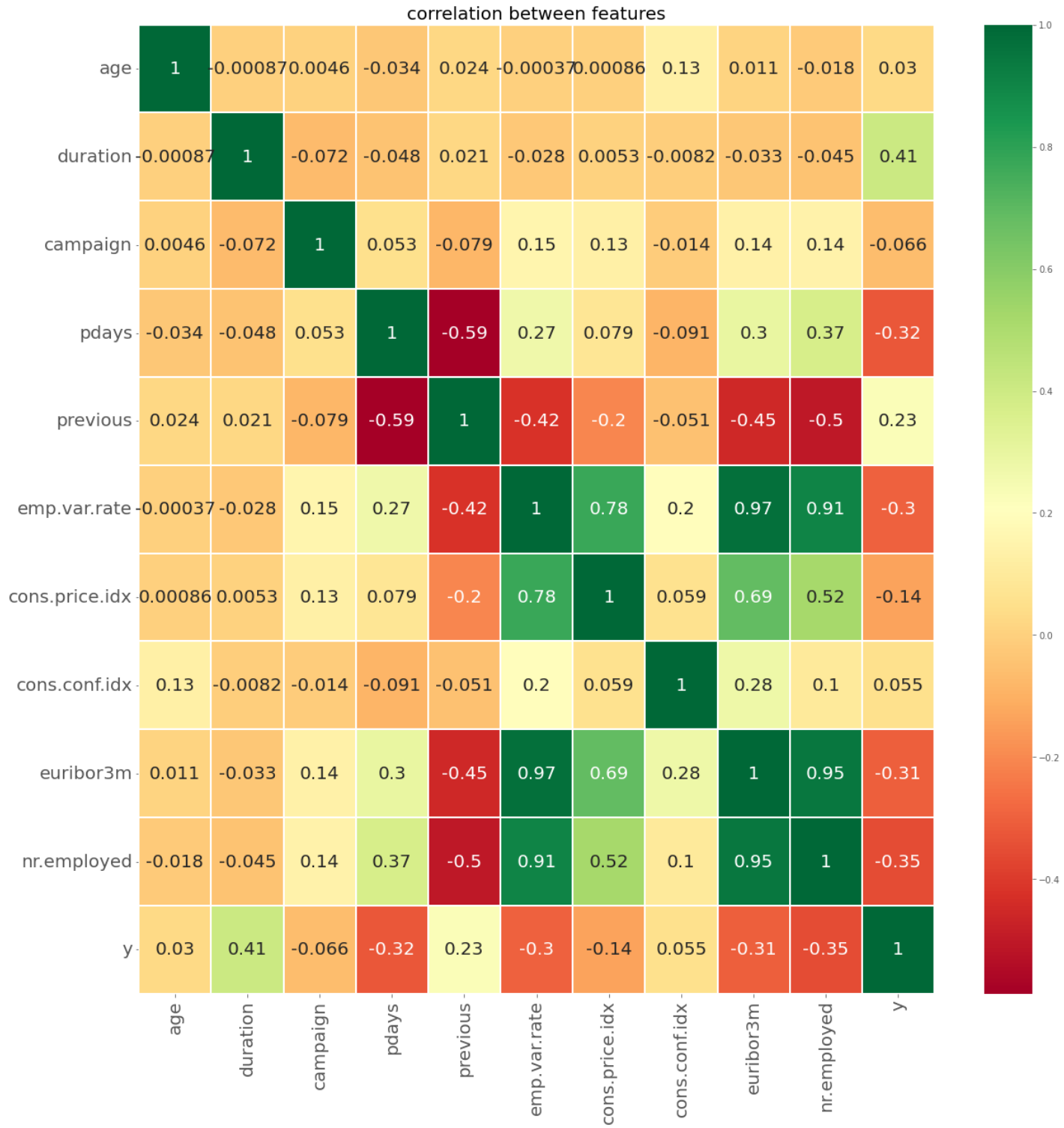
In [24]:
```
train['y']=train['y'].replace('no',0)
train['y']=train['y'].replace('yes',1)
```

In [25]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41188 entries, 0 to 41187
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   age             41188 non-null  int64  
 1   job             41188 non-null  object 
 2   marital         41188 non-null  object 
 3   education       41188 non-null  object 
 4   default         41188 non-null  object 
 5   housing         41188 non-null  object 
 6   loan            41188 non-null  object 
 7   contact         41188 non-null  object 
 8   month           41188 non-null  object 
 9   day_of_week     41188 non-null  object 
 10  duration        41188 non-null  int64  
 11  campaign        41188 non-null  int64  
 12  pdays           41188 non-null  int64  
 13  previous        41188 non-null  int64  
 14  poutcome        41188 non-null  object 
 15  emp.var.rate    41188 non-null  float64
 16  cons.price.idx  41188 non-null  float64
 17  cons.conf.idx   41188 non-null  float64
 18  euribor3m       41188 non-null  float64
 19  nr.employed     41188 non-null  float64
 20  y               41188 non-null  int64  
dtypes: float64(5), int64(6), object(10)
memory usage: 6.6+ MB
```

In [27]:
```
heatmap = sns.heatmap(train[['age','duration','campaign','pdays','previous','emp.var.rate'
                             'euribor3m','nr.employed','y']].corr(),annot = True, cmap='Rd
fig = plt.gcf()
fig.set_size_inches(20,20)
heatmap.set_xticklabels(heatmap.get_xticklabels(),fontsize=20)
heatmap.set_yticklabels(heatmap.get_yticklabels(), fontsize=20)

plt.title('correlation between features', fontsize=20)
plt.show()
#신기하게  senseor number -4 와 어떤 음의 상관관계를 가진다.
#상관관계를 가지기에, pca를 통해 차원 축소가 가능해보인다.
#줄이면, 간단한 svm도 가능하지 않을까?
```



가설에서 생각한것 처럼 16~20는 주변 상황을 의미하는 변수로, 선형 변환으로 하나로 그룹 되지 않을까? (변수 가공

높은 선형성을 보인다.

높은 상관성을 보이는 피처는 duration, previous,pdays ,emp.var.rate, cons.price.idx , euribor3m,nr.employed

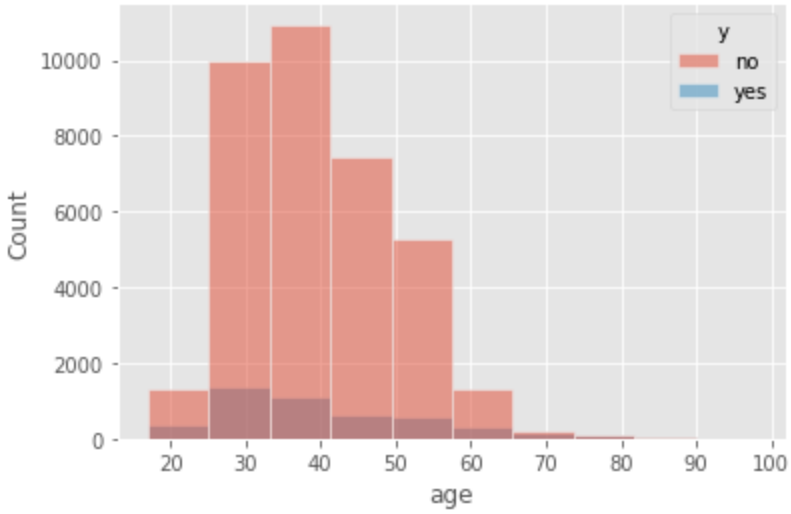
social and economic context attributes과 이전 접촉과관련 변수들이 가장 높았다.

age,campaign, cons.conf.idx는 사용 안할 변수로 보인다.

age

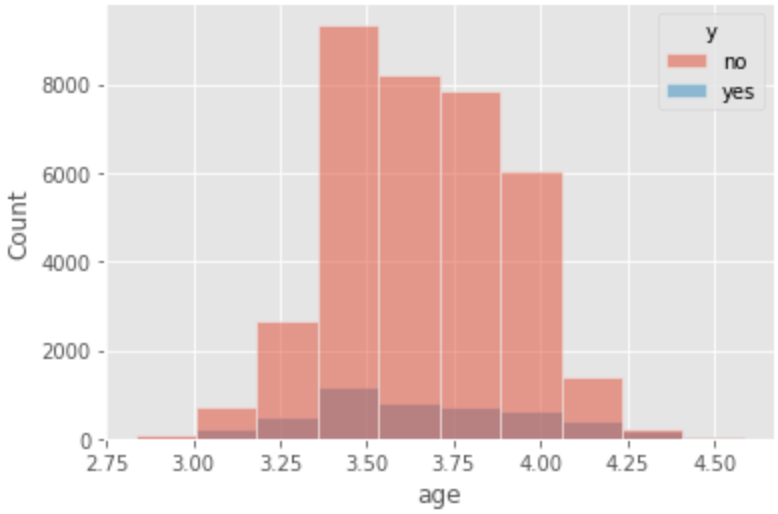
In [18]:
```
sns.histplot(x='age',hue='y',bins=10,data=train)
```

Out[18]: <AxesSubplot:xlabel='age', ylabel='Count'>



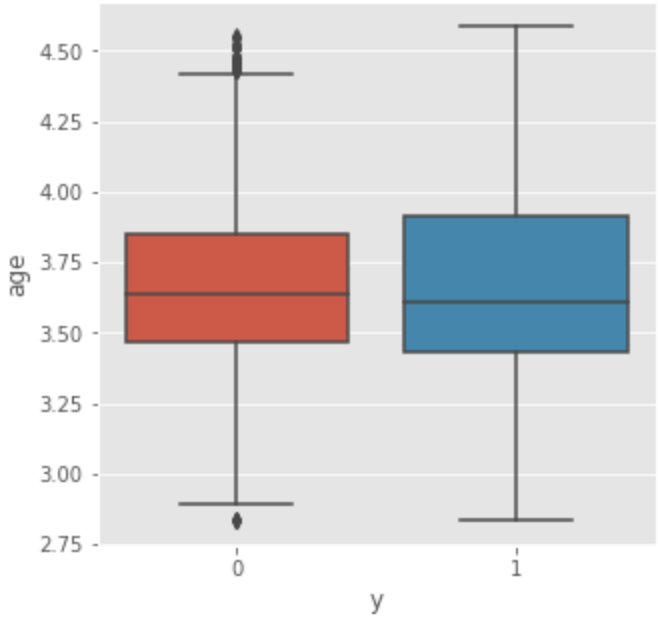
In [19]:
```
sns.histplot(x=np.log(train['age']),hue=train['y'],bins=10)
```

Out[19]: <AxesSubplot:xlabel='age', ylabel='Count'>



In [32]:
```
plt.figure(figsize=(5,5))
sns.boxplot(data=train,y=np.log(train['age']),x=train['y'])
#target=1에서 age hour가 중앙값이 더 크게 형성된다.
```

Out[32]: <AxesSubplot:xlabel='y', ylabel='age'>



나이는 별로 유의성이 없어보인다. 비교적 25~35가 가장 많아보인다. 도수가 가장 높기 때문.

In [28]:
```
kstest(train['age'], train['y'])
```

Out[28]:
```
KstestResult(statistic=1.0, pvalue=0.0)
```

이미지에서는 별로 상관성이 없어보이는데 여기서는 유의하게 나온다.

나이대에 따라 비율이 다를 수 있기 때문인가.

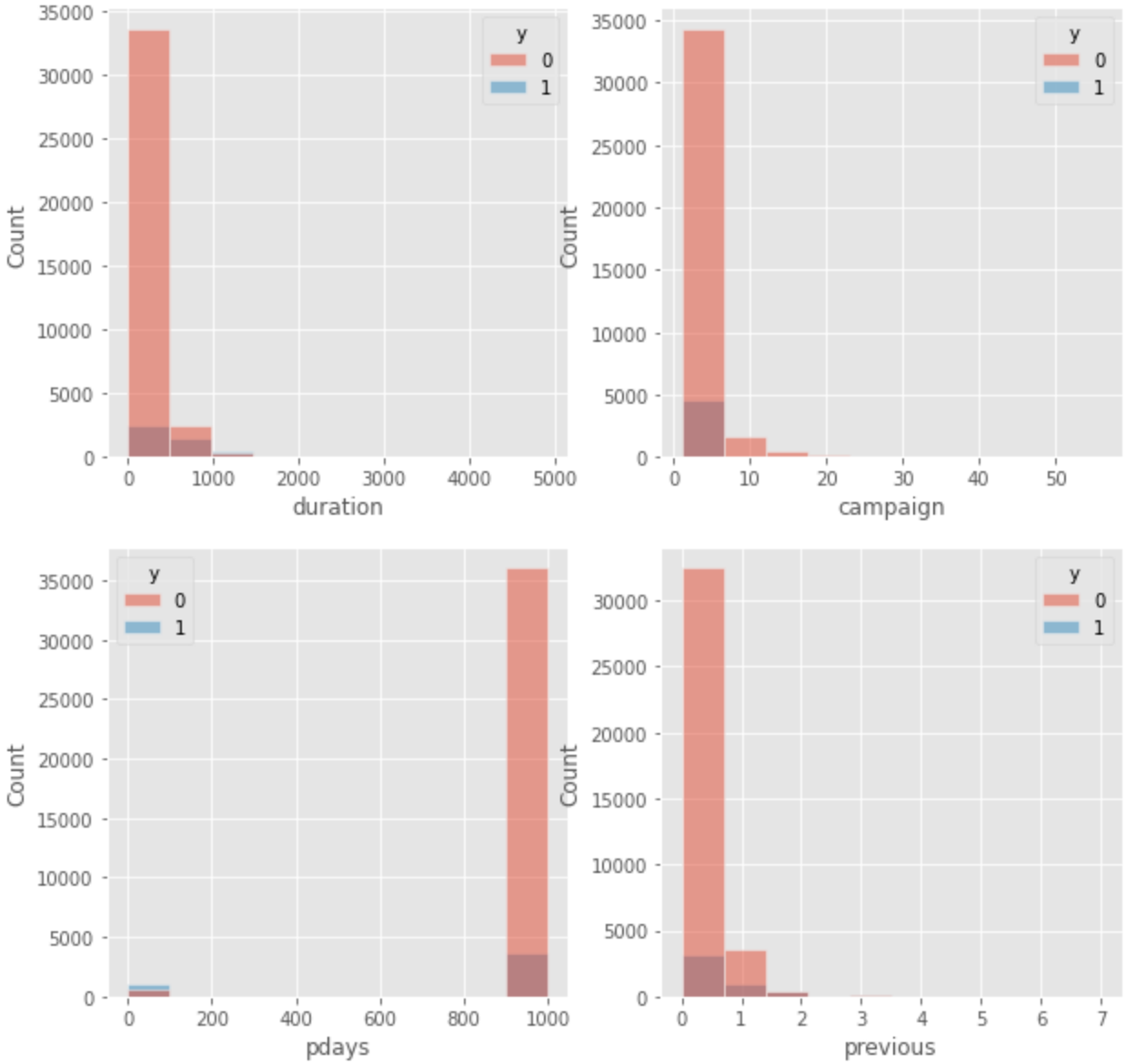
사용하지 않는것이 좋아보인다.

duration & campaign & pdays & previous

이전 접촉과 관련된 자료

In [38]:
```
plt.figure(figsize=(10,10))
plt.subplot(2,2,1)
sns.histplot(x='duration',hue='y',bins=10,data=train)
plt.subplot(2,2,2)
sns.histplot(x='campaign',hue='y',bins=10,data=train)
plt.subplot(2,2,3)
sns.histplot(x='pdays',hue='y',bins=10,data=train)
plt.subplot(2,2,4)
sns.histplot(x='previous',hue='y',bins=10,data=train)
```

Out[38]:
```
<AxesSubplot:xlabel='previous', ylabel='Count'>
```

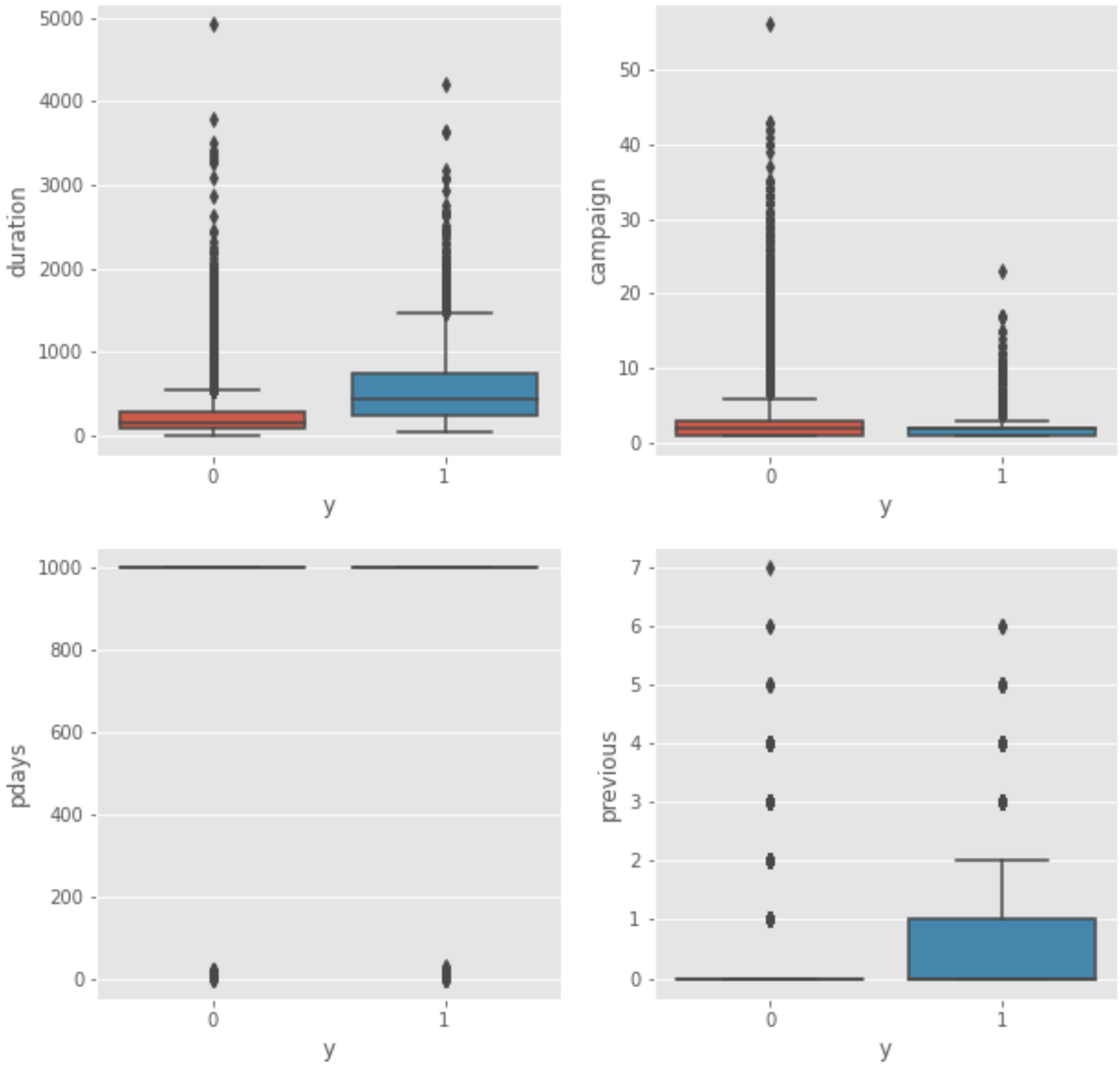


In [44]:

```
plt.figure(figsize=(10,10))
plt.subplot(2,2,1)
sns.boxplot(data=train,y=train['duration'],x=train['y'])
plt.subplot(2,2,2)
sns.boxplot(data=train,y=train['campaign'],x=train['y'])
plt.subplot(2,2,3)
sns.boxplot(data=train,y=train['pdays'],x=train['y'])
plt.subplot(2,2,4)
sns.boxplot(data=train,y=train['previous'],x=train['y'])
```

Out[44]:

```
<AxesSubplot:xlabel='y', ylabel='previous'>
```



- duaration은 커질수록 비율이 높아지는 경향이 있고
- campaign은 낮을 수록 유의하다
- pdays는 999는 접촉하지 않았음을 의미한다. 그래서 범주형 변수로 변환해야한다.
- previous는 유의한지 판단이 어렵다.

In [40]:
```
print("duration : ",kstest(train['duration'], train['y']))
print("campaign : ",kstest(train['campaign'], train['y']))
print("pdays : ",kstest(train['pdays'], train['y']))
print("previous : ",kstest(train['previous'], train['y']))
```

```
duration :  KstestResult(statistic=0.9998300475866757, pvalue=0.0)
campaign :  KstestResult(statistic=0.8873458288821987, pvalue=0.0)
pdays :  KstestResult(statistic=0.9990045644362435, pvalue=0.0)
previous :  KstestResult(statistic=0.0258327668252889, pvalue=2.263920670210774e-12)
```

기각하며, 유의하다고 할 수 있다.

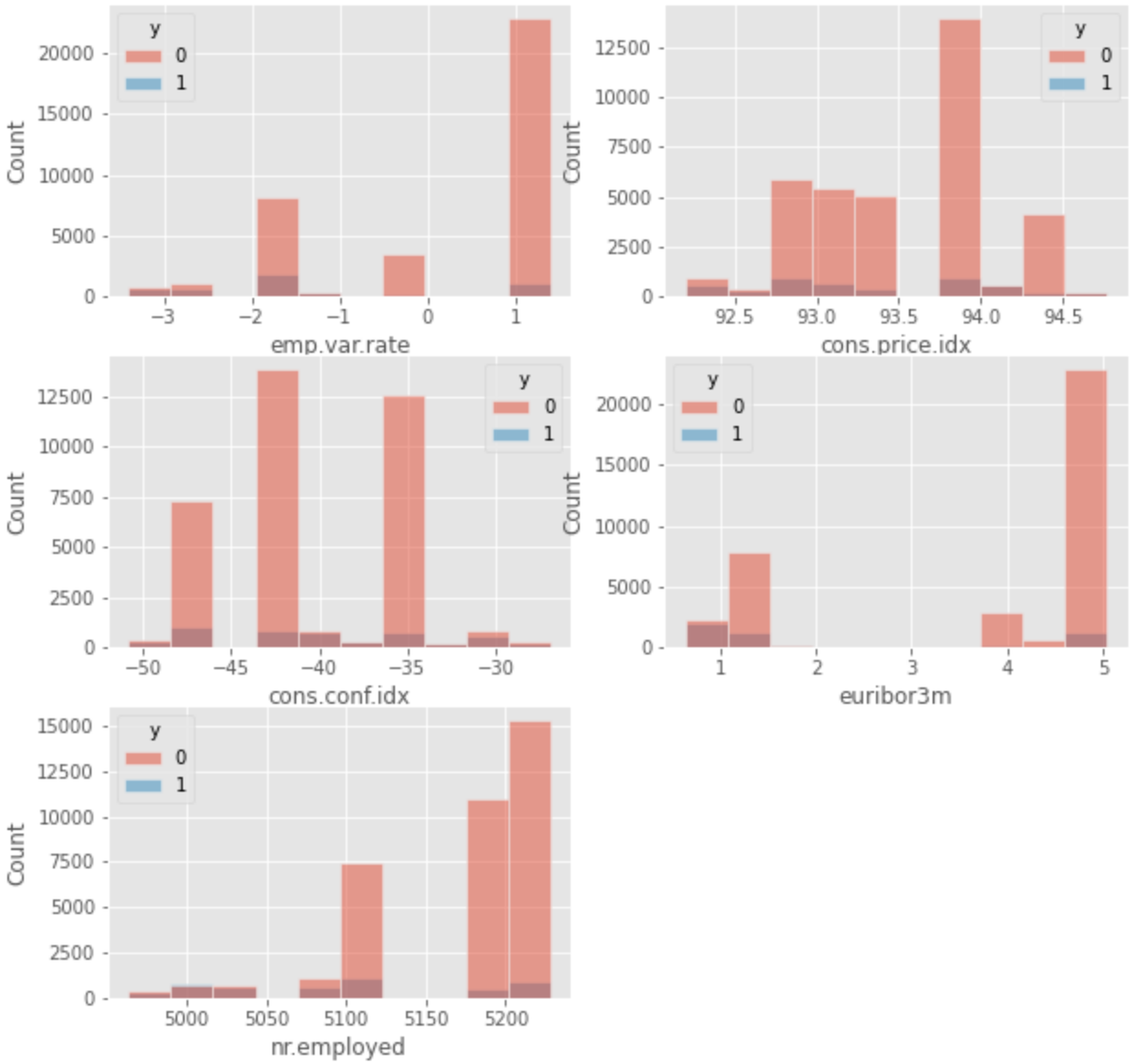
변수들을 모두 사용한다.

social and economic context attributes

emp.var.rate, cons.price.idx, cons.conf.idx, euribor3m, nr.employed 변수들을 분석

In [42]:
```
plt.figure(figsize=(10,10))
plt.subplot(3,2,1)
sns.histplot(x='emp.var.rate',hue='y',bins=10,data=train)
plt.subplot(3,2,2)
sns.histplot(x='cons.price.idx',hue='y',bins=10,data=train)
plt.subplot(3,2,3)
sns.histplot(x='cons.conf.idx',hue='y',bins=10,data=train)
plt.subplot(3,2,4)
sns.histplot(x='euribor3m',hue='y',bins=10,data=train)
plt.subplot(3,2,5)
sns.histplot(x='nr.employed',hue='y',bins=10,data=train)
```

Out[42]:
```
<AxesSubplot:xlabel='nr.employed', ylabel='Count'>
```



데이터가 연속되지 않는 경우가 많다.

박스플롯 사용은 안된다.

In [45]:
```
print("emp.var.rate : ",kstest(train['emp.var.rate'], train['y']))
print("cons.price.idx : ",kstest(train['cons.price.idx'], train['y']))
print("cons.conf.idx : ",kstest(train['cons.conf.idx'], train['y']))
print("euribor3m : ",kstest(train['euribor3m'], train['y']))
print("nr.employed : ",kstest(train['nr.employed'], train['y']))
```

```
emp.var.rate :  KstestResult(statistic=0.5826211517917841, pvalue=0.0)
cons.price.idx :  KstestResult(statistic=1.0, pvalue=0.0)
cons.conf.idx :  KstestResult(statistic=1.0, pvalue=0.0)
```

```
euribor3m :  KstestResult(statistic=0.9051179955326795, pvalue=0.0)
nr.employed :  KstestResult(statistic=1.0, pvalue=0.0)
```

모두 유의하다는 결론이 나온다.

변수 가공

위에서 찾은 변수와 target의 관계를 통해 변환을 한다.

In [60]:
```
train_change = train.copy()
```

In [61]:
```
# job grouping

train_change['job']=train_change['job'].replace(['blue-collar','services','entrepreneur',
                            'unknown','management','admin.','unemployed'],'job1')
train_change['job'].unique()
```

Out[61]:
```
array(['job1', 'retired', 'student'], dtype=object)
```

In [62]:
```
# marital grouping

train_change['marital']=train_change['marital'].replace(['married','divorced'],'marital1')
train_change['marital']=train_change['marital'].replace(['single','unknown'],'marital2')
train_change['marital'].unique()
```

Out[62]:
```
array(['marital1', 'marital2'], dtype=object)
```

In [63]:
```
# education grouping

train_change['education']=train_change['education'].replace(['basic.9y','basic.6y','basic.
train_change['education']=train_change['education'].replace(['university.degree','unknown'
train_change['education'].unique()
```

Out[63]:
```
array(['education1', 'professional.course', 'university.degree',
       'illiterate'], dtype=object)
```

In [64]:
```
# default grouping

train_change['default']=train_change['default'].replace(['unknown'],'yes')
train_change['default'].unique()
```

Out[64]:
```
array(['no', 'yes'], dtype=object)
```

In [65]:
```
# housing & loan 제거

train_change.drop(['housing','loan'],axis=1,inplace=True)
```

In [66]:
```
# month grouping
# may / jul,nov,jun, aug / apr / oct,sep / dec,mar/ 로 구분할 수 있다.

train_change['month']=train_change['month'].replace(['jul','nov','jun','aug'],'month1')
train_change['month']=train_change['month'].replace(['oct','sep'],'month2')
train_change['month']=train_change['month'].replace(['dec','mar'],'month3')
train_change['month'].unique()
```

Out[66]:
```
array(['may', 'month1', 'month2', 'month3', 'apr'], dtype=object)
```

In [67]:
```
# day_of_week grouping
#mon,fri / tue,wed,thu 으로 구분이된다.

train_change['day_of_week']=train_change['day_of_week'].replace(['mon','fri'],'day1')
train_change['day_of_week']=train_change['day_of_week'].replace(['tue','wed','thu'],'day2'
train_change['day_of_week'].unique()
```

Out[67]:
```
array(['day1', 'day2'], dtype=object)
```

In [68]:
```
# age 제거
train_change.drop('age',axis=1,inplace=True)
```

In [79]:
```
# pdays 범주형 변환
# 만났으면 1 , 만나지 않았으면 0으로 분류.

train_change['pdays']=train_change['pdays'].where(train_change['pdays'] !=999, 1)
train_change['pdays']=train_change['pdays'].where(train_change['pdays'] ==999, 0)
```

In [80]:
```
train_change
```

Out[80]:

	job	marital	education	default	contact	month	day_of_week	duration	campaign	pdays	p
0	job1	marital1	education1	no	telephone	may	day1	261	1	0	
1	job1	marital1	education1	yes	telephone	may	day1	149	1	0	
2	job1	marital1	education1	no	telephone	may	day1	226	1	0	
3	job1	marital1	education1	no	telephone	may	day1	151	1	0	
4	job1	marital1	education1	no	telephone	may	day1	307	1	0	
...	...	...	...	...	...	...	...	...	...	...	
41183	retired	marital1	professional.course	no	cellular	month1	day1	334	1	0	
41184	job1	marital1	professional.course	no	cellular	month1	day1	383	1	0	
41185	retired	marital1	university.degree	no	cellular	month1	day1	189	2	0	
41186	job1	marital1	professional.course	no	cellular	month1	day1	442	1	0	
41187	retired	marital1	professional.course	no	cellular	month1	day1	239	3	0	

41188 rows × 18 columns

In [81]:
```
train_change.to_csv('./bankfull_eda.csv',index=False)
```

결론

각 변수들이 주는 영향에 따라 변환을 해주었고, 결과는 다음과 같다.

- 분포에 따라 job을 grouping 해주었다. (3가지 범주)
- 분포에 따라 marital을 grouping 해주었다. (2가지 범주)
- 분포에 따라 education grouping 해주었다. (4가지 범주)

- default는 unknonw을 신용불량자로 변환해주었다.
- housing과 loan은 차이가 유의하지 않기에, 변수 제거. (변수제거)
- month는 may / jul,nov,jun, aug / apr / oct,sep / dec,mar/ 로 구분할 수 있다. (5가지 범주)
- day_of_week는 mon,fri / tue,wed,thu 으로 구분이된다.
- age 변수는 group간에 차이가 없어보여서, 제거.
- padys를 범주형으로 변환 해주었다. (만났으면1, 만났지 않았으면 0)