In [1]:

```python
try:
    from IPython import get_ipython
    get_ipython().magic('clear')
    get_ipython().magic('reset -f')
except:
    pass

#Importing all the required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import missingno as msno
from scipy.stats import zscore as zscore
#To ignore the warnings
import warnings as wg
wg.filterwarnings("ignore")

# Set plot style
sns.set(color_codes=True)

# Set maximum number of columns to be displayed
pd.set_option('display.max_columns', 100)


#Read the dataset file of train data
train_data=pd.read_csv("ml_case_training_data.csv")
#Read the dataset file of history data
history_data=pd.read_csv("ml_case_training_hist_data.csv")
#Read the dataset file of churn data
churn_data=pd.read_csv("ml_case_training_output.csv")

train = pd.merge(train_data, churn_data, on="id")
```

In [2]:

```python
train.head()
```

Out[2]:

	id	activity_new	campaign_disc_ele	channel_sales	cons_12m	cons_gas_12m	cons_last_month	date_activ	date_end	date_fi
0	48ada52261e7cf58715202705a0451c9	esoiiifxdlbkcsluxmfuacbdckommixw	NaN	lmkebamcaaclubfxadlmueccxoimlema	309275	0	10025	2012-11-07	2016-11-06	
1	24011ae4ebbe3035111d65fa7c15bc57	NaN	NaN	foosdfpfkusacimwkcsosbicdxkicaua	0	54946	0	2013-06-15	2016-06-15	
2	d29c2c54acc38ff3c0614d0a653813dd	NaN	NaN	NaN	4660	0	0	2009-08-21	2016-08-30	
3	764c75f661154dac3a6c254cd082ea7d	NaN	NaN	foosdfpfkusacimwkcsosbicdxkicaua	544	0	0	2010-04-16	2016-04-16	
4	bba03439a292a1e166f80264c16191cb	NaN	NaN	lmkebamcaaclubfxadlmueccxoimlema	1584	0	0	2010-03-30	2016-03-30	

In [3]:

```python
pd.DataFrame({"Data type":train.dtypes})
```

Out[3]:

	Data type
id	object
activity_new	object
campaign_disc_ele	float64
channel_sales	object
cons_12m	int64
cons_gas_12m	int64
cons_last_month	int64
date_activ	object
date_end	object
date_first_activ	object
date_modif_prod	object
date_renewal	object
forecast_base_bill_ele	float64
forecast_base_bill_year	float64
forecast_bill_12m	float64
forecast_cons	float64
forecast_cons_12m	float64
forecast_cons_year	int64
forecast_discount_energy	float64
forecast_meter_rent_12m	float64
forecast_price_energy_p1	float64
forecast_price_energy_p2	float64
forecast_price_pow_p1	float64
has_gas	object
imp_cons	float64
margin_gross_pow_ele	float64
margin_net_pow_ele	float64
nb_prod_act	int64
net_margin	float64
num_years_antig	int64
origin_up	object
pow_max	float64
churn	int64

In [4]:

```python
pd.DataFrame({"Data type":history_data.dtypes})
```

Out[4]:

	Data type
id	object
price_date	object
price_p1_var	float64
price_p2_var	float64
price_p3_var	float64
price_p1_fix	float64
price_p2_fix	float64
price_p3_fix	float64

In [5]:

```python
train.describe()
```

Out[5]:

	campaign_disc_ele	cons_12m	cons_gas_12m	cons_last_month	forecast_base_bill_ele	forecast_base_bill_year	forecast_bill_12m	forecast_cons	forecast_cons_12m	forecast_cons_year	for
count	0.0	1.609600e+04	1.609600e+04	1.609600e+04	3508.000000	3508.000000	3508.000000	3508.000000	16096.000000	16096.000000	
mean	NaN	1.948044e+05	3.191164e+04	1.946154e+04	335.843857	335.843857	3837.441866	206.845165	2370.555949	1907.347229	
std	NaN	6.795151e+05	1.775885e+05	8.235676e+04	649.406000	649.406000	5425.744327	455.634288	4035.085664	5257.364759	
min	NaN	-1.252760e+05	-3.037000e+03	-9.138600e+04	-364.940000	-364.940000	-2503.480000	0.000000	-16689.260000	-85627.000000	
25%	NaN	5.906250e+03	0.000000e+00	0.000000e+00	0.000000	0.000000	1158.175000	0.000000	513.230000	0.000000	
50%	NaN	1.533250e+04	0.000000e+00	9.010000e+02	162.955000	162.955000	2187.230000	42.215000	1179.160000	378.000000	
75%	NaN	5.022150e+04	0.000000e+00	4.127000e+03	396.185000	396.185000	4246.555000	228.117500	2692.077500	1994.250000	
max	NaN	1.609711e+07	4.188440e+06	4.538720e+06	12566.080000	12566.080000	81122.630000	9682.890000	103801.930000	175375.000000	

In [6]:

```python
train["campaign_disc_ele"].isnull().values.all()
```

Out[6]:

```
True
```

In [7]:

```python
history_data.describe()
```

Out[7]:

	price_p1_var	price_p2_var	price_p3_var	price_p1_fix	price_p2_fix	price_p3_fix
count	191643.000000	191643.000000	191643.000000	191643.000000	191643.000000	191643.000000
mean	0.140991	0.054412	0.030712	43.325546	10.698201	6.455436
std	0.025117	0.050033	0.036335	5.437952	12.856046	7.782279
min	0.000000	0.000000	0.000000	-0.177779	-0.097752	-0.065172
25%	0.125976	0.000000	0.000000	40.728885	0.000000	0.000000
50%	0.146033	0.085483	0.000000	44.266930	0.000000	0.000000
75%	0.151635	0.101780	0.072558	44.444710	24.339581	16.226389
max	0.280700	0.229788	0.114102	59.444710	36.490692	17.458221

In [8]:

```python
pd.DataFrame({"Missing values (%)": train.isnull().sum()/len(train.index)*100})
```

Out[8]:

	Missing values (%)
id	0.000000
activity_new	59.300447
campaign_disc_ele	100.000000
channel_sales	26.205268
cons_12m	0.000000
cons_gas_12m	0.000000
cons_last_month	0.000000
date_activ	0.000000
date_end	0.012425
date_first_activ	78.205765
date_modif_prod	0.975398
date_renewal	0.248509
forecast_base_bill_ele	78.205765
forecast_base_bill_year	78.205765
forecast_bill_12m	78.205765
forecast_cons	78.205765
forecast_cons_12m	0.000000
forecast_cons_year	0.000000
forecast_discount_energy	0.782803
forecast_meter_rent_12m	0.000000
forecast_price_energy_p1	0.782803
forecast_price_energy_p2	0.782803
forecast_price_pow_p1	0.782803
has_gas	0.000000
imp_cons	0.000000
margin_gross_pow_ele	0.080765
margin_net_pow_ele	0.080765
nb_prod_act	0.000000
net_margin	0.093191
num_years_antig	0.000000
origin_up	0.540507
pow_max	0.018638
churn	0.000000

In [9]:

```python
train.drop(['campaign_disc_ele','date_first_activ','forecast_base_bill_ele','forecast_cons','forecast_bill_12m','forecast_base_bill_year'],inplace=True,axis=1
```

In [10]:

```python
df=train.head()
```

In [11]:

```python
def impute_nan(df,variable,median):
    df[variable]=df[variable].fillna(median)

median=df.net_margin.median()

median

impute_nan(df,'forecast_discount_energy',median)
impute_nan(df,'forecast_price_energy_p1',median)
impute_nan(df,'forecast_price_energy_p2',median)
impute_nan(df,'forecast_price_pow_p1',median)
impute_nan(df,'margin_gross_pow_ele',median)
impute_nan(df,'margin_net_pow_ele',median)
impute_nan(df,'net_margin',median)
impute_nan(df,'pow_max',median)

def impute_nan(df,variable):
    most_frequent_category=df[variable].mode()[0]
    df[variable].fillna(most_frequent_category,inplace=True)

for feature in ['activity_new','channel_sales','date_modif_prod']:
    impute_nan(df,feature)

df.isnull().sum()
```

Out[11]:

```
id                          0
activity_new                0
channel_sales               0
cons_12m                    0
cons_gas_12m                0
cons_last_month             0
date_activ                  0
date_end                    0
date_modif_prod             0
date_renewal                0
forecast_cons_12m           0
forecast_cons_year          0
forecast_discount_energy    0
forecast_meter_rent_12m     0
forecast_price_energy_p1    0
forecast_price_energy_p2    0
forecast_price_pow_p1       0
has_gas                     0
imp_cons                    0
margin_gross_pow_ele        0
margin_net_pow_ele          0
nb_prod_act                 0
net_margin                  0
num_years_antig             0
origin_up                   0
pow_max                     0
churn                       0
dtype: int64
```

In [12]:

```python
# fill missing values with mean column values
history_data.fillna(history_data.mean(), inplace=True)
# count the number of NaN values in each column
print(history_data.isnull().sum())
```

```
id              0
price_date      0
price_p1_var    0
price_p2_var    0
price_p3_var    0
price_p1_fix    0
price_p2_fix    0
price_p3_fix    0
dtype: int64
```

In [13]:

```python
train=pd.merge(df,history_data,on='id')
```

In [14]:

```python
pd.DataFrame({"Dataframe columns": train.columns})
```

Out[14]:

	Dataframe columns
0	id
1	activity_new
2	channel_sales
3	cons_12m
4	cons_gas_12m
5	cons_last_month
6	date_activ
7	date_end
8	date_modif_prod
9	date_renewal
10	forecast_cons_12m
11	forecast_cons_year
12	forecast_discount_energy
13	forecast_meter_rent_12m
14	forecast_price_energy_p1
15	forecast_price_energy_p2
16	forecast_price_pow_p1
17	has_gas
18	imp_cons
19	margin_gross_pow_ele
20	margin_net_pow_ele
21	nb_prod_act
22	net_margin
23	num_years_antig
24	origin_up
25	pow_max
26	churn
27	price_date
28	price_p1_var
29	price_p2_var
30	price_p3_var
31	price_p1_fix
32	price_p2_fix
33	price_p3_fix

In [15]:

```python
train.dtypes
```

Out[15]:

```
id                           object
activity_new                 object
channel_sales                object
cons_12m                      int64
cons_gas_12m                  int64
cons_last_month               int64
date_activ                   object
date_end                     object
date_modif_prod              object
date_renewal                 object
forecast_cons_12m           float64
forecast_cons_year            int64
forecast_discount_energy    float64
forecast_meter_rent_12m     float64
forecast_price_energy_p1    float64
forecast_price_energy_p2    float64
forecast_price_pow_p1       float64
has_gas                      object
imp_cons                    float64
margin_gross_pow_ele        float64
margin_net_pow_ele          float64
nb_prod_act                   int64
net_margin                  float64
num_years_antig               int64
origin_up                    object
pow_max                     float64
churn                         int64
price_date                   object
price_p1_var                float64
price_p2_var                float64
price_p3_var                float64
price_p1_fix                float64
price_p2_fix                float64
price_p3_fix                float64
dtype: object
```

In [16]:

```python
train.head()
```

Out[16]:

	id	activity_new	channel_sales	cons_12m	cons_gas_12m	cons_last_month	date_activ	date_end	date_modif_prod	date_rene
0	48ada52261e7cf58715202705a0451c9	esoiiifxdlbkcsluxmfuacbdckommixw	lmkebamcaaclubfxadlmueccxoimlema	309275	0	10025	2012-11-07	2016-11-06	2012-11-07	2015-1
1	48ada52261e7cf58715202705a0451c9	esoiiifxdlbkcsluxmfuacbdckommixw	lmkebamcaaclubfxadlmueccxoimlema	309275	0	10025	2012-11-07	2016-11-06	2012-11-07	2015-1
2	48ada52261e7cf58715202705a0451c9	esoiiifxdlbkcsluxmfuacbdckommixw	lmkebamcaaclubfxadlmueccxoimlema	309275	0	10025	2012-11-07	2016-11-06	2012-11-07	2015-1
3	48ada52261e7cf58715202705a0451c9	esoiiifxdlbkcsluxmfuacbdckommixw	lmkebamcaaclubfxadlmueccxoimlema	309275	0	10025	2012-11-07	2016-11-06	2012-11-07	2015-1
4	48ada52261e7cf58715202705a0451c9	esoiiifxdlbkcsluxmfuacbdckommixw	lmkebamcaaclubfxadlmueccxoimlema	309275	0	10025	2012-11-07	2016-11-06	2012-11-07	2015-1

In [17]:

```python
def change_into_datetime(col):
    train[col]=pd.to_datetime(train[col])
```

In [18]:

```python
train.columns
```

Out[18]:

```
Index(['id', 'activity_new', 'channel_sales', 'cons_12m', 'cons_gas_12m',
       'cons_last_month', 'date_activ', 'date_end', 'date_modif_prod',
       'date_renewal', 'forecast_cons_12m', 'forecast_cons_year',
       'forecast_discount_energy', 'forecast_meter_rent_12m',
       'forecast_price_energy_p1', 'forecast_price_energy_p2',
       'forecast_price_pow_p1', 'has_gas', 'imp_cons', 'margin_gross_pow_ele',
       'margin_net_pow_ele', 'nb_prod_act', 'net_margin', 'num_years_antig',
       'origin_up', 'pow_max', 'churn', 'price_date', 'price_p1_var',
       'price_p2_var', 'price_p3_var', 'price_p1_fix', 'price_p2_fix',
       'price_p3_fix'],
      dtype='object')
```

In [19]:

```python
for i in ['date_activ','date_end', 'price_date','date_renewal']:
    change_into_datetime(i)
```

In [20]:

```python
train.dtypes
```

Out[20]:

```
id                                  object
activity_new                        object
channel_sales                       object
cons_12m                             int64
cons_gas_12m                         int64
cons_last_month                      int64
date_activ                  datetime64[ns]
date_end                    datetime64[ns]
date_modif_prod                     object
date_renewal                datetime64[ns]
forecast_cons_12m                  float64
forecast_cons_year                   int64
forecast_discount_energy           float64
forecast_meter_rent_12m            float64
forecast_price_energy_p1           float64
forecast_price_energy_p2           float64
forecast_price_pow_p1              float64
has_gas                             object
imp_cons                           float64
margin_gross_pow_ele               float64
margin_net_pow_ele                 float64
nb_prod_act                          int64
net_margin                         float64
num_years_antig                      int64
origin_up                           object
pow_max                            float64
churn                                int64
price_date                  datetime64[ns]
price_p1_var                       float64
price_p2_var                       float64
price_p3_var                       float64
price_p1_fix                       float64
price_p2_fix                       float64
price_p3_fix                       float64
dtype: object
```

In [21]:

```python
train['date_activ_day']=train['date_activ'].dt.day
train['date_activ_month']=train['date_activ'].dt.month
train['date_activ_year']=train['date_activ'].dt.year
```

In [22]:

```python
train['date_end_day']=train['date_end'].dt.day
train['date_end_month']=train['date_end'].dt.month
train['date_end_year']=train['date_end'].dt.year
```

In [23]:

```python
train['price_date_day']=train['price_date'].dt.day
train['price_date_month']=train['price_date'].dt.month
train['price_date_year']=train['price_date'].dt.year
```

In [24]:

```python
train['date_renewal_day']=train['date_renewal'].dt.day
train['date_renewal_month']=train['date_renewal'].dt.month
train['date_renewal_year']=train['date_renewal'].dt.year
#train['date_modif_prod_day']=train['date_modif_prod'].dt.day
#train['date_modif_prod_month']=train['date_modif_prod'].dt.month
#train['date_modif_prod_year']=train['date_modif_prod'].dt.year
```

In [25]:

```python
train.dtypes
```

Out[25]:

```
id                                  object
activity_new                        object
channel_sales                       object
cons_12m                             int64
cons_gas_12m                         int64
cons_last_month                      int64
date_activ                  datetime64[ns]
date_end                    datetime64[ns]
date_modif_prod                     object
date_renewal                datetime64[ns]
forecast_cons_12m                  float64
forecast_cons_year                   int64
forecast_discount_energy           float64
forecast_meter_rent_12m            float64
forecast_price_energy_p1           float64
forecast_price_energy_p2           float64
forecast_price_pow_p1              float64
has_gas                             object
imp_cons                           float64
margin_gross_pow_ele               float64
margin_net_pow_ele                 float64
nb_prod_act                          int64
net_margin                         float64
num_years_antig                      int64
origin_up                           object
pow_max                            float64
churn                                int64
price_date                  datetime64[ns]
price_p1_var                       float64
price_p2_var                       float64
price_p3_var                       float64
price_p1_fix                       float64
price_p2_fix                       float64
price_p3_fix                       float64
date_activ_day                       int64
date_activ_month                     int64
date_activ_year                      int64
date_end_day                         int64
date_end_month                       int64
date_end_year                        int64
price_date_day                       int64
price_date_month                     int64
price_date_year                      int64
date_renewal_day                     int64
date_renewal_month                   int64
date_renewal_year                    int64
dtype: object
```

In [26]:

```python
train.drop("activity_new",inplace=True,axis=1)
train.drop("channel_sales",inplace=True,axis=1)
train.drop("origin_up",inplace=True,axis=1)
train.drop("date_end",inplace=True,axis=1)
train.drop("date_activ",inplace=True,axis=1)
train.drop("price_date",inplace=True,axis=1)
train.drop("date_renewal",inplace=True,axis=1)
train.drop("date_modif_prod",inplace=True,axis=1)
train.drop("has_gas",inplace=True,axis=1)
```

In [27]:

```python
Y = train["churn"]
X = train.drop(labels = ["id","churn"],axis = 1)
```

In [28]:

```python
# Split the dataset into 75% Training set and 25% Testing set
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.25, random_state = 0)
```

In [29]:

```python
!pip install xgboost
```

```
Requirement already satisfied: xgboost in c:\users\pooja\anaconda3\lib\site-packages (1.4.2)
Requirement already satisfied: numpy in c:\users\pooja\anaconda3\lib\site-packages (from xgboost) (1.19.2)
Requirement already satisfied: scipy in c:\users\pooja\anaconda3\lib\site-packages (from xgboost) (1.5.2)
WARNING: You are using pip version 21.1.1; however, version 21.2.3 is available.
You should consider upgrading via the 'c:\users\pooja\anaconda3\python.exe -m pip install --upgrade pip' command.
```

In [39]:

```python
#Use Random Forest Classification algorithm
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.model_selection import StratifiedKFold
import xgboost as xgb
model = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 0)
result = model.fit(X_train,Y_train)
```

In [40]:

```python
def evaluate(model_, X_test_, Y_test_):
    prediction_test_ = model_.predict(X_test)
    results = pd.DataFrame({"Accuracy" : [metrics.accuracy_score(Y_test_, prediction_test_)],
                            "Precision" : [metrics.precision_score(Y_test_, prediction_test_)],
                            "Recall" : [metrics.recall_score(Y_test_, prediction_test_)]})
    return results
```

In [41]:

```python
evaluate(model, X_test, Y_test)
```

Out[41]:

	Accuracy	Precision	Recall
0	1.0	1.0	1.0