BCGX Churn Analysis

---

1. Import packages
2. Load data
3. Feature engineering

---

1. Import packages

In [1…
```
import pandas as pd
import matplotlib.pyplot as plt
import pandas as pd
import pylab as pl
import seaborn as sns
import numpy as np
import warnings
warnings.filterwarnings("ignore")
```

---

2. Load data

In [2…
```
client_df = pd.read_csv('client_data.csv')
price_df = pd.read_csv('price_data.csv')

client_df["date_activ"] = pd.to_datetime(client_df["date_acti
client_df["date_end"] = pd.to_datetime(client_df["date_end"],
client_df["date_modif_prod"] = pd.to_datetime(client_df["date
client_df["date_renewal"] = pd.to_datetime(client_df["date_re
price_df['price_date'] = pd.to_datetime(price_df['price_date'
```

In [3…
```
analysed_data=client_df.merge(price_df, how ='inner', on=['id
```

In [4…
```
analysed_data.to_csv("analysed_data.csv")
```

In [5…
```
df = pd.read_csv('./analysed_data.csv')
df["date_activ"] = pd.to_datetime(df["date_activ"], format='%
df["date_end"] = pd.to_datetime(df["date_end"], format='%Y-%m
df["date_modif_prod"] = pd.to_datetime(df["date_modif_prod"],
df["date_renewal"] = pd.to_datetime(df["date_renewal"], forma
```

In [6…
```
df.head(3)
```

Out[6]:

	Unnamed: 0	id	channe
0	0	24011ae4ebbe3035111d65fa7c15bc57	foosdfpfkusacimwkcsosbicd
1	1	24011ae4ebbe3035111d65fa7c15bc57	foosdfpfkusacimwkcsosbicd
2	2	24011ae4ebbe3035111d65fa7c15bc57	foosdfpfkusacimwkcsosbicd

3 rows × 34 columns

4.Data Cleaning

---

3. Feature engineering

Difference between off-peak prices in December and preceding January

Below is the code created by your colleague to calculate the feature described above. Use this code to re-create this feature and then think about ways to build on this feature to create features with a higher predictive power.

In [7…
```
df.columns
```

Out[7]:
```
Index(['Unnamed: 0', 'id', 'channel_sales', 'cons_12m', 'con
s_gas_12m',
       'cons_last_month', 'date_activ', 'date_end', 'date_mo
dif_prod',
       'date_renewal', 'forecast_cons_12m', 'forecast_cons_y
ear',
       'forecast_discount_energy', 'forecast_meter_rent_12
m',
       'forecast_price_energy_off_peak', 'forecast_price_ene
rgy_peak',
       'forecast_price_pow_off_peak', 'has_gas', 'imp_cons',
       'margin_gross_pow_ele', 'margin_net_pow_ele', 'nb_pro
d_act',
       'net_margin', 'num_years_antig', 'origin_up', 'pow_ma
x', 'churn',
       'price_date', 'price_off_peak_var', 'price_peak_var',
       'price_mid_peak_var', 'price_off_peak_fix', 'price_pe
ak_fix',
       'price_mid_peak_fix'],
      dtype='object')
```

In [8…
```
df = df.dropna()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 175149 entries, 0 to 175148
Data columns (total 34 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   Unnamed: 0                      175149 non-null  int64
 1   id                              175149 non-null  object
 2   channel_sales                   175149 non-null  object
 3   cons_12m                        175149 non-null  int64
 4   cons_gas_12m                    175149 non-null  int64
 5   cons_last_month                 175149 non-null  int64
 6   date_activ                      175149 non-null  dateti
me64[ns]
 7   date_end                        175149 non-null  dateti
me64[ns]
 8   date_modif_prod                 175149 non-null  dateti
me64[ns]
 9   date_renewal                    175149 non-null  dateti
me64[ns]
 10  forecast_cons_12m               175149 non-null  float6
4
 11  forecast_cons_year              175149 non-null  int64
 12  forecast_discount_energy        175149 non-null  float6
4
```

```
 13   forecast_meter_rent_12m         175149 non-null  float6
4
 14   forecast_price_energy_off_peak  175149 non-null  float6
4
 15   forecast_price_energy_peak      175149 non-null  float6
4
 16   forecast_price_pow_off_peak     175149 non-null  float6
4
 17   has_gas                         175149 non-null  object
 18   imp_cons                        175149 non-null  float6
4
 19   margin_gross_pow_ele            175149 non-null  float6
4
 20   margin_net_pow_ele              175149 non-null  float6
4
 21   nb_prod_act                     175149 non-null  int64
 22   net_margin                      175149 non-null  float6
4
 23   num_years_antig                 175149 non-null  int64
 24   origin_up                       175149 non-null  object
 25   pow_max                         175149 non-null  float6
4
 26   churn                           175149 non-null  int64
 27   price_date                      175149 non-null  object
 28   price_off_peak_var              175149 non-null  float6
4
 29   price_peak_var                  175149 non-null  float6
4
 30   price_mid_peak_var              175149 non-null  float6
4
 31   price_off_peak_fix              175149 non-null  float6
4
 32   price_peak_fix                  175149 non-null  float6
4
 33   price_mid_peak_fix              175149 non-null  float6
4
dtypes: datetime64[ns](4), float64(17), int64(8), object(5)
memory usage: 45.4+ MB
```

In [9…

```
price_df = pd.read_csv('price_data.csv')
price_df["price_date"] = pd.to_datetime(price_df["price_date"
price_df.head()
```

Out[9]:

	id	price_date	price_off_peak_var	pric
0	038af19179925da21a25619c5a24b745	2015-01-01	0.151367	
1	038af19179925da21a25619c5a24b745	2015-02-01	0.151367	
2	038af19179925da21a25619c5a24b745	2015-03-01	0.151367	
3	038af19179925da21a25619c5a24b745	2015-04-01	0.149626	
4	038af19179925da21a25619c5a24b745	2015-05-01	0.149626	

In […

In […

In [1…

```
# Group off-peak prices by companies and month
monthly_price_by_id = price_df.groupby(['id', 'price_date']).

# Get january and december prices
jan_prices = monthly_price_by_id.groupby('id').first().reset_
dec_prices = monthly_price_by_id.groupby('id').last().reset_i

# Calculate the difference
diff = pd.merge(dec_prices.rename(columns={'price_off_peak_va
diff['offpeak_diff_dec_january_energy'] = diff['dec_1'] - dif
diff['offpeak_diff_dec_january_power'] = diff['dec_2'] - diff
diff = diff[['id', 'offpeak_diff_dec_january_energy','offpeak
diff.head()
```

Out[10]:

	id	offpeak_diff_dec_january_energy
0	0002203ffbb812588b632b9e628cc38d	-0.006192
1	0004351ebdd665e6ee664792efc4fd13	-0.004104
2	0010bcc39e42b3c2131ed2ce55246e3c	0.050443
3	0010ee3855fdea87602a5b7aba8e42de	-0.010018
4	00114d74e963e47177db89bc70108537	-0.003994

In […

In [1…
```
offpeak_diff_dec_january_energy=diff['offpeak_diff_dec_januar
offpeak_diff_dec_january_power=diff['offpeak_diff_dec_january
df['offpeak_diff_dec_january_energy']= offpeak_diff_dec_janua
df['offpeak_diff_dec_january_power']= offpeak_diff_dec_januar
```

Adding the Month and Year the Subscription ended Feature

This is necessary as it will help us to further understand which months have the most churns. With this data we can know which years and months had the most churned customers.

In [1…
```
month_ended= df['date_end'].dt.month
year_ended= df['date_end'].dt.year
df['month_sub_ended']=month_ended
df['year_sub_ended']=year_ended
```

In […

In […

Adding the Month and Year the Subscription Began Feature

This is necessary as it will help us to further understand which months have the most churns. With this data we can know which years and months had the most subscriptions.

In [1…
```
month_activ= df['date_activ'].dt.month
year_active= df['date_activ'].dt.year
df['month_sub_began']=month_activ
df['year_sub_began']=year_active
```

In […

Adding the number of years a customer was subscribed

This will help us know how long a customer had a subscription

In [1…
```
Num_of_sub_years= df['year_sub_ended'] - df['year_sub_began']
Num_of_sub_years
```

Out[14]:
```
0         3
1         3
2         3
3         3
4         3
         ..
175144    7
175145    7
175146    7
175147    7
175148    7
Length: 175149, dtype: int32
```

In [1…
```
Num_of_sub_years.unique()
```

Out[15]:
```
array([ 3,  7,  6,  5,  4, 11, 12,  9, 10,  8, 13,  2], dty
pe=int32)
```

Adding Num_of_sub_years to the dataframe

In [1…
```
df['Num_of_sub_years']=Num_of_sub_years
```

In [1…
```
df = df.dropna()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 16096 entries, 0 to 16095
Data columns (total 41 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Unnamed: 0                      16096 non-null  int64
 1   id                              16096 non-null  object
 2   channel_sales                   16096 non-null  object
 3   cons_12m                        16096 non-null  int64
 4   cons_gas_12m                    16096 non-null  int64
 5   cons_last_month                 16096 non-null  int64
 6   date_activ                      16096 non-null  dateti
me64[ns]
```

```
 7   date_end                        16096 non-null  dateti
me64[ns]
 8   date_modif_prod                 16096 non-null  dateti
me64[ns]
 9   date_renewal                    16096 non-null  dateti
me64[ns]
 10  forecast_cons_12m               16096 non-null  float6
4
 11  forecast_cons_year              16096 non-null  int64
 12  forecast_discount_energy        16096 non-null  float6
4
 13  forecast_meter_rent_12m         16096 non-null  float6
4
 14  forecast_price_energy_off_peak  16096 non-null  float6
4
 15  forecast_price_energy_peak      16096 non-null  float6
4
 16  forecast_price_pow_off_peak     16096 non-null  float6
4
 17  has_gas                         16096 non-null  object
 18  imp_cons                        16096 non-null  float6
4
 19  margin_gross_pow_ele            16096 non-null  float6
4
 20  margin_net_pow_ele              16096 non-null  float6
4
 21  nb_prod_act                     16096 non-null  int64
 22  net_margin                      16096 non-null  float6
4
 23  num_years_antig                 16096 non-null  int64
 24  origin_up                       16096 non-null  object
 25  pow_max                         16096 non-null  float6
4
 26  churn                           16096 non-null  int64
 27  price_date                      16096 non-null  object
 28  price_off_peak_var              16096 non-null  float6
4
 29  price_peak_var                  16096 non-null  float6
4
 30  price_mid_peak_var              16096 non-null  float6
4
 31  price_off_peak_fix              16096 non-null  float6
4
 32  price_peak_fix                  16096 non-null  float6
4
 33  price_mid_peak_fix              16096 non-null  float6
4
 34  offpeak_diff_dec_january_energy 16096 non-null  float6
```

```
4
 35  offpeak_diff_dec_january_power   16096 non-null  float6
4
 36  month_sub_ended                  16096 non-null  int32
 37  year_sub_ended                   16096 non-null  int32
 38  month_sub_began                  16096 non-null  int32
 39  year_sub_began                   16096 non-null  int32
 40  Num_of_sub_years                 16096 non-null  int32
dtypes: datetime64[ns](4), float64(19), int32(5), int64(8),
object(5)
memory usage: 4.9+ MB
```

In [1…

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 16096 entries, 0 to 16095
Data columns (total 41 columns):
 #   Column                           Non-Null Count  Dtype
---  ------                           --------------  -----
 0   Unnamed: 0                       16096 non-null  int64
 1   id                               16096 non-null  object
 2   channel_sales                    16096 non-null  object
 3   cons_12m                         16096 non-null  int64
 4   cons_gas_12m                     16096 non-null  int64
 5   cons_last_month                  16096 non-null  int64
 6   date_activ                       16096 non-null  dateti
me64[ns]
 7   date_end                         16096 non-null  dateti
me64[ns]
 8   date_modif_prod                  16096 non-null  dateti
me64[ns]
 9   date_renewal                     16096 non-null  dateti
me64[ns]
 10  forecast_cons_12m                16096 non-null  float6
4
 11  forecast_cons_year               16096 non-null  int64
 12  forecast_discount_energy         16096 non-null  float6
4
 13  forecast_meter_rent_12m          16096 non-null  float6
4
 14  forecast_price_energy_off_peak   16096 non-null  float6
4
 15  forecast_price_energy_peak       16096 non-null  float6
4
 16  forecast_price_pow_off_peak      16096 non-null  float6
4
 17  has_gas                          16096 non-null  object
 18  imp_cons                         16096 non-null  float6
```

```
4
 19  margin_gross_pow_ele            16096 non-null  float6
4
 20  margin_net_pow_ele              16096 non-null  float6
4
 21  nb_prod_act                     16096 non-null  int64
 22  net_margin                      16096 non-null  float6
4
 23  num_years_antig                 16096 non-null  int64
 24  origin_up                       16096 non-null  object
 25  pow_max                         16096 non-null  float6
4
 26  churn                           16096 non-null  int64
 27  price_date                      16096 non-null  object
 28  price_off_peak_var              16096 non-null  float6
4
 29  price_peak_var                  16096 non-null  float6
4
 30  price_mid_peak_var              16096 non-null  float6
4
 31  price_off_peak_fix              16096 non-null  float6
4
 32  price_peak_fix                  16096 non-null  float6
4
 33  price_mid_peak_fix              16096 non-null  float6
4
 34  offpeak_diff_dec_january_energy  16096 non-null  float6
4
 35  offpeak_diff_dec_january_power  16096 non-null  float6
4
 36  month_sub_ended                 16096 non-null  int32
 37  year_sub_ended                  16096 non-null  int32
 38  month_sub_began                 16096 non-null  int32
 39  year_sub_began                  16096 non-null  int32
 40  Num_of_sub_years                16096 non-null  int32
dtypes: datetime64[ns](4), float64(19), int32(5), int64(8),
object(5)
memory usage: 4.9+ MB
```

In […

```
```

In [1…

```
df=df.drop(columns=['Unnamed: 0','date_activ','date_end','dat
```

We will transform features, channel_sales, has_gas , origin_up

In [2…

```
df
```

Out[20]:

	id	channel_sale
0	24011ae4ebbe3035111d65fa7c15bc57	foosdfpfkusacimwkcsosbicdxkicau
1	24011ae4ebbe3035111d65fa7c15bc57	foosdfpfkusacimwkcsosbicdxkicau
2	24011ae4ebbe3035111d65fa7c15bc57	foosdfpfkusacimwkcsosbicdxkicau
3	24011ae4ebbe3035111d65fa7c15bc57	foosdfpfkusacimwkcsosbicdxkicau
4	24011ae4ebbe3035111d65fa7c15bc57	foosdfpfkusacimwkcsosbicdxkicau
...	...	
16091	0082e565c1298cbe9cf70e96f571e4fa	MISSIN
16092	e84c8b7d0e7e31cfebb07f46ac7445cf	usilxuppasemubllopkaafesmlibms
16093	e84c8b7d0e7e31cfebb07f46ac7445cf	usilxuppasemubllopkaafesmlibms
16094	e84c8b7d0e7e31cfebb07f46ac7445cf	usilxuppasemubllopkaafesmlibms
16095	e84c8b7d0e7e31cfebb07f46ac7445cf	usilxuppasemubllopkaafesmlibms

16096 rows × 35 columns

In [2…

```
df['channel_sales']=df['channel_sales'].astype('category')
```

In [2…

```
df['channel_sales'].unique
```

Out[22]:

```
<bound method Series.unique of 0        foosdfpfkusacimwkcs
osbicdxkicaua
1        foosdfpfkusacimwkcsosbicdxkicaua
2        foosdfpfkusacimwkcsosbicdxkicaua
3        foosdfpfkusacimwkcsosbicdxkicaua
4        foosdfpfkusacimwkcsosbicdxkicaua
                       ...
16091                             MISSING
16092    usilxuppasemubllopkaafesmlibmsdf
16093    usilxuppasemubllopkaafesmlibmsdf
16094    usilxuppasemubllopkaafesmlibmsdf
16095    usilxuppasemubllopkaafesmlibmsdf
Name: channel_sales, Length: 16096, dtype: category
Categories (6, object): ['MISSING', 'epumfxlbckeskwekxbiuas
klxalciiuu', 'ewpakwlliwisiwduibdlfmalxowmwpci', 'foosdfpfk
usacimwkcsosbicdxkicaua', 'lmkebamcaaclubfxadlmueccxoimlem
a', 'usilxuppasemubllopkaafesmlibmsdf']>
```

In [2…
```
df = pd.get_dummies(df, columns=['channel_sales'], prefix='ch
```

In [2…
```
df
```

Out[24]:

	id	cons_12m	cons_gas_12m	co
0	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
1	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
2	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
3	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
4	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
...	...	...	...	
16091	0082e565c1298cbe9cf70e96f571e4fa	9779	351	
16092	e84c8b7d0e7e31cfebb07f46ac7445cf	7273	186456	
16093	e84c8b7d0e7e31cfebb07f46ac7445cf	7273	186456	
16094	e84c8b7d0e7e31cfebb07f46ac7445cf	7273	186456	
16095	e84c8b7d0e7e31cfebb07f46ac7445cf	7273	186456	

16096 rows × 40 columns

In [2…
```
df['channel_foosdfpfkusacimwkcsosbicdxkicaua'] = df['channel_
df['channel_lmkebamcaaclubfxadlmueccxoimlema'] = df['channel_
df['channel_usilxuppasemubllopkaafesmlibmsdf'] = df['channel_
df['channel_epumfxlbckeskwekxbiuasklxalciiuu'] = df['channel_
df['channel_ewpakwlliwisiwduibdlfmalxowmwpci'] = df['channel_
df['channel_MISSING'] = df['channel_MISSING'].astype(int)
```

In […

In [2…
```
df['origin_up'].value_counts()
```

Out[26]:
```
origin_up
lxidpiddsbxsbosboudacockeimpuepw    8008
kamkkxfxxuwbdslkwifmmcsiusiuosws    4660
ldkssxwpmemidmecebumciepifcamkci    3320
MISSING                               96
usapbepcfoloekilkwsdiboslwaxobdp      12
Name: count, dtype: int64
```

In [2…

```
df['origin_up']=df['origin_up'].astype('category')
df = pd.get_dummies(df, columns=['origin_up'], prefix='origin
```

In [2…

```
df.head(5)
```

Out[28]:

	id	cons_12m	cons_gas_12m	cons_las
0	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
1	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
2	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
3	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
4	24011ae4ebbe3035111d65fa7c15bc57	0	54946	

5 rows × 44 columns

In [2…

```
df['has_gas']=df['has_gas'].replace(['t','f'],[1,0])
```

In [3…

```
df['origin_MISSING'] = df['origin_MISSING'].astype(int)
df['origin_kamkkxfxxuwbdslkwifmmcsiusiuosws'] = df['origin_ka
df['origin_ldkssxwpmemidmecebumciepifcamkci'] = df['origin_ld
df['origin_lxidpiddsbxsbosboudacockeimpuepw'] = df['origin_lx
df['origin_usapbepcfoloekilkwsdiboslwaxobdp'] = df['origin_us
```

In [3…

```
df=df.drop(columns=['origin_MISSING'])
```

In [3…

```
df=df.drop(columns=['origin_usapbepcfoloekilkwsdiboslwaxobdp'
```

In [3…

```
df.head(5)
```

Out[33]:

	id	cons_12m	cons_gas_12m	cons_las
0	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
1	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
2	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
3	24011ae4ebbe3035111d65fa7c15bc57	0	54946	
4	24011ae4ebbe3035111d65fa7c15bc57	0	54946	

5 rows × 42 columns

In [3…

```
df.describe()
```

Out[34]:

	cons_12m	cons_gas_12m	cons_last_month	forecast_cons_12n
count	1.609600e+04	1.609600e+04	16096.000000	16096.00000(
mean	1.845098e+05	2.719006e+04	18905.320017	1865.80430‹
std	6.764478e+05	1.621541e+05	73714.458229	2070.7756O:
min	0.000000e+00	0.000000e+00	0.000000	0.00000(
25%	5.443000e+03	0.000000e+00	0.000000	506.30000(
50%	1.370400e+04	0.000000e+00	716.500000	1155.13000(
75%	3.768800e+04	0.000000e+00	3218.000000	2497.09000(
max	6.207104e+06	1.959386e+06	771203.000000	18481.68000(

8 rows × 41 columns

In […

```

```

Looking at the standard deviation from df we are able to tell which features are skewed

In [3…

```
skewed=["cons_12m","cons_gas_12m","cons_last_month","forecast
```

In [3…

```
df[skewed].describe()
```

Out[36]:

	cons_12m	cons_gas_12m	cons_last_month	forecast_cons_12n
count	1.609600e+04	1.609600e+04	16096.000000	16096.00000(
mean	1.845098e+05	2.719006e+04	18905.320017	1865.80430
std	6.764478e+05	1.621541e+05	73714.458229	2070.77560
min	0.000000e+00	0.000000e+00	0.000000	0.00000(
25%	5.443000e+03	0.000000e+00	0.000000	506.30000(
50%	1.370400e+04	0.000000e+00	716.500000	1155.13000(
75%	3.768800e+04	0.000000e+00	3218.000000	2497.09000(
max	6.207104e+06	1.959386e+06	771203.000000	18481.68000(

Transformation of skewed data

In [3…

```
df["cons_12m"] = np.log10(df["cons_12m"] + 1)
df["cons_gas_12m"] = np.log10(df["cons_gas_12m"] + 1)
df["cons_last_month"] = np.log10(df["cons_last_month"] + 1)
df["forecast_cons_12m"] = np.log10(df["forecast_cons_12m"] +
df["forecast_cons_year"] = np.log10(df["forecast_cons_year"]
df["forecast_meter_rent_12m"] = np.log10(df["forecast_meter_r
df["forecast_discount_energy"] = np.log10(df["forecast_discou
df["net_margin"] = np.log10(df["net_margin"] + 1)
df["num_years_antig"] = np.log10(df["num_years_antig"] + 1)
df["pow_max"] = np.log10(df["pow_max"] + 1)
df["margin_gross_pow_ele"] = np.log10(df["margin_gross_pow_el
df["margin_net_pow_ele"] = np.log10(df["margin_net_pow_ele"]
df["price_off_peak_fix"] = np.log10(df["price_off_peak_fix"]
df["price_peak_fix"] = np.log10(df["price_peak_fix"] + 1)
df["price_mid_peak_fix"] = np.log10(df["price_mid_peak_fix"]
df["Num_of_sub_years"] = np.log10(df["Num_of_sub_years"] + 1)
df["year_sub_began"] = np.log10(df["year_sub_began"] + 1)
```

In [3…

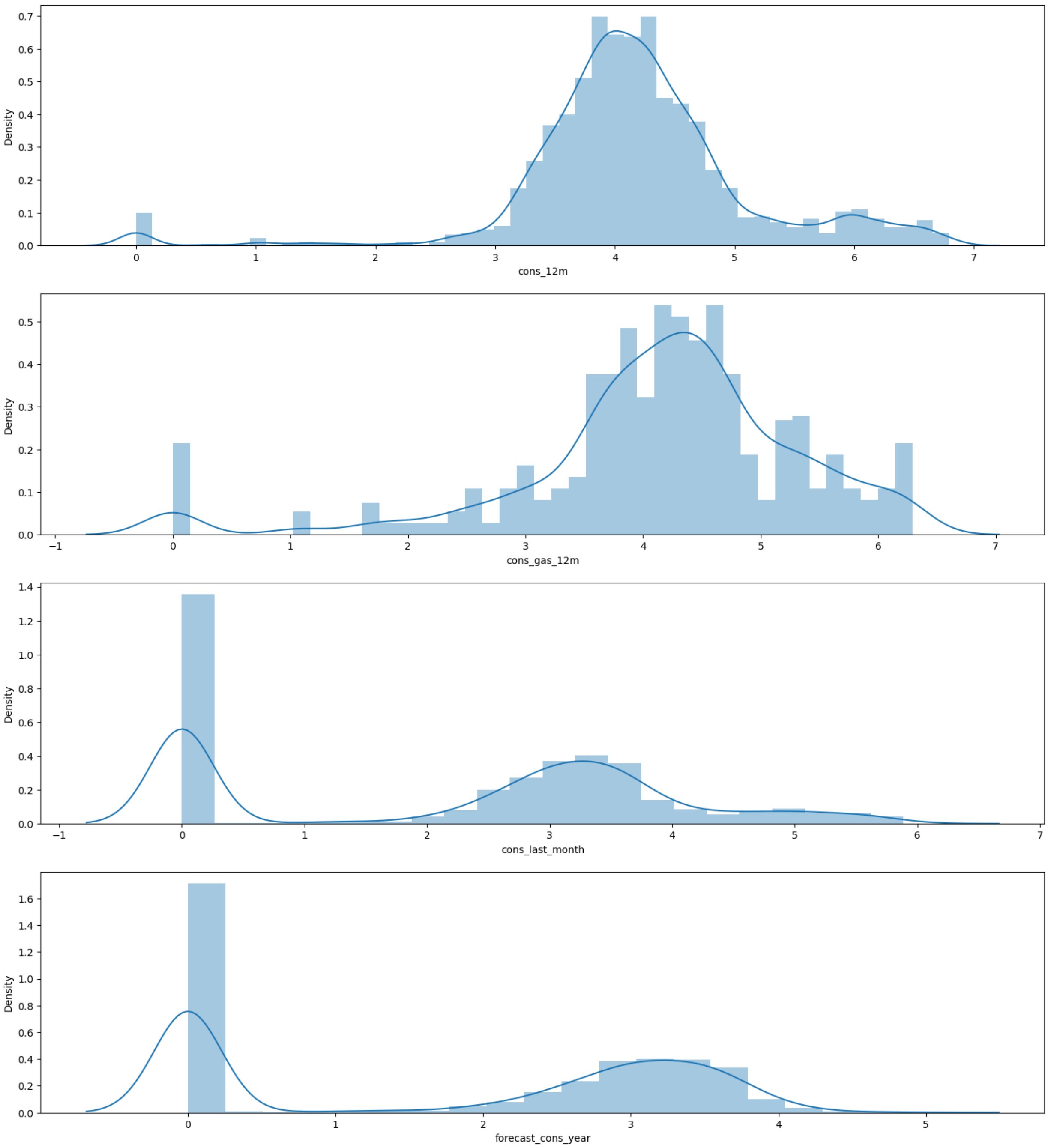
```
df[skewed].describe()
```

Out[38]:

	cons_12m	cons_gas_12m	cons_last_month	forecast_cons_12m
count	16096.000000	16096.000000	16096.000000	16096.000000
mean	4.189114	0.800043	2.194268	2.960570
std	0.959502	1.724448	1.801889	0.705170
min	0.000000	0.000000	0.000000	0.000000
25%	3.735918	0.000000	0.000000	2.705265
50%	4.136879	0.000000	2.855817	3.063007
75%	4.576215	0.000000	3.507721	3.397608
max	6.792889	6.292120	5.887169	4.266765

In [3…

```
fig, axs = plt.subplots(nrows=4, figsize=(18, 20))
# Plot histograms
sns.distplot((df["cons_12m"].dropna()), ax=axs[0])
sns.distplot((df[df["has_gas"]==1]["cons_gas_12m"].dropna()),
sns.distplot((df["cons_last_month"].dropna()), ax=axs[2])
sns.distplot((df["forecast_cons_year"].dropna()), ax=axs[3])
plt.show()
```



Features cons_gas_12m, cons_last_month, forecast_cons_year still show some level of skewness

In [4…

```
df[skewed].describe()
```

Out[40]:

	cons_12m	cons_gas_12m	cons_last_month	forecast_cons_12m
count	16096.000000	16096.000000	16096.000000	16096.00000
mean	4.189114	0.800043	2.194268	2.96057
std	0.959502	1.724448	1.801889	0.70517
min	0.000000	0.000000	0.000000	0.00000
25%	3.735918	0.000000	0.000000	2.70526
50%	4.136879	0.000000	2.855817	3.06300
75%	4.576215	0.000000	3.507721	3.39760
max	6.792889	6.292120	5.887169	4.26676

In […

In [4…

```
df.columns
```

Out[41]:
```
Index(['id', 'cons_12m', 'cons_gas_12m', 'cons_last_month',
       'forecast_cons_12m', 'forecast_cons_year', 'forecast
_discount_energy',
       'forecast_meter_rent_12m', 'forecast_price_energy_of
f_peak',
       'forecast_price_energy_peak', 'forecast_price_pow_of
f_peak', 'has_gas',
       'imp_cons', 'margin_gross_pow_ele', 'margin_net_pow_
ele', 'nb_prod_act',
       'net_margin', 'num_years_antig', 'pow_max', 'churn',
       'price_off_peak_var', 'price_peak_var', 'price_mid_p
eak_var',
       'price_off_peak_fix', 'price_peak_fix', 'price_mid_p
eak_fix',
       'offpeak_diff_dec_january_energy', 'offpeak_diff_dec
_january_power',
       'month_sub_ended', 'year_sub_ended', 'month_sub_bega
n',
       'year_sub_began', 'Num_of_sub_years', 'channel_MISSI
NG',
       'channel_epumfxlbckeskwekxbiuasklxalciiuu',
       'channel_ewpakwlliwisiwduibdlfmalxowmwpci',
       'channel_foosdfpfkusacimwkcsosbicdxkicaua',
       'channel_lmkebamcaaclubfxadlmueccxoimlema',
       'channel_usilxuppasemubllopkaafesmlibmsdf',
       'origin_kamkkxfxxuwbdslkwifmmcsiusiuosws',
       'origin_ldkssxwpmemidmecebumciepifcamkci',
       'origin_lxidpiddsbxsbosboudacockeimpuepw'],
      dtype='object')
```

In [4…
```
df
```

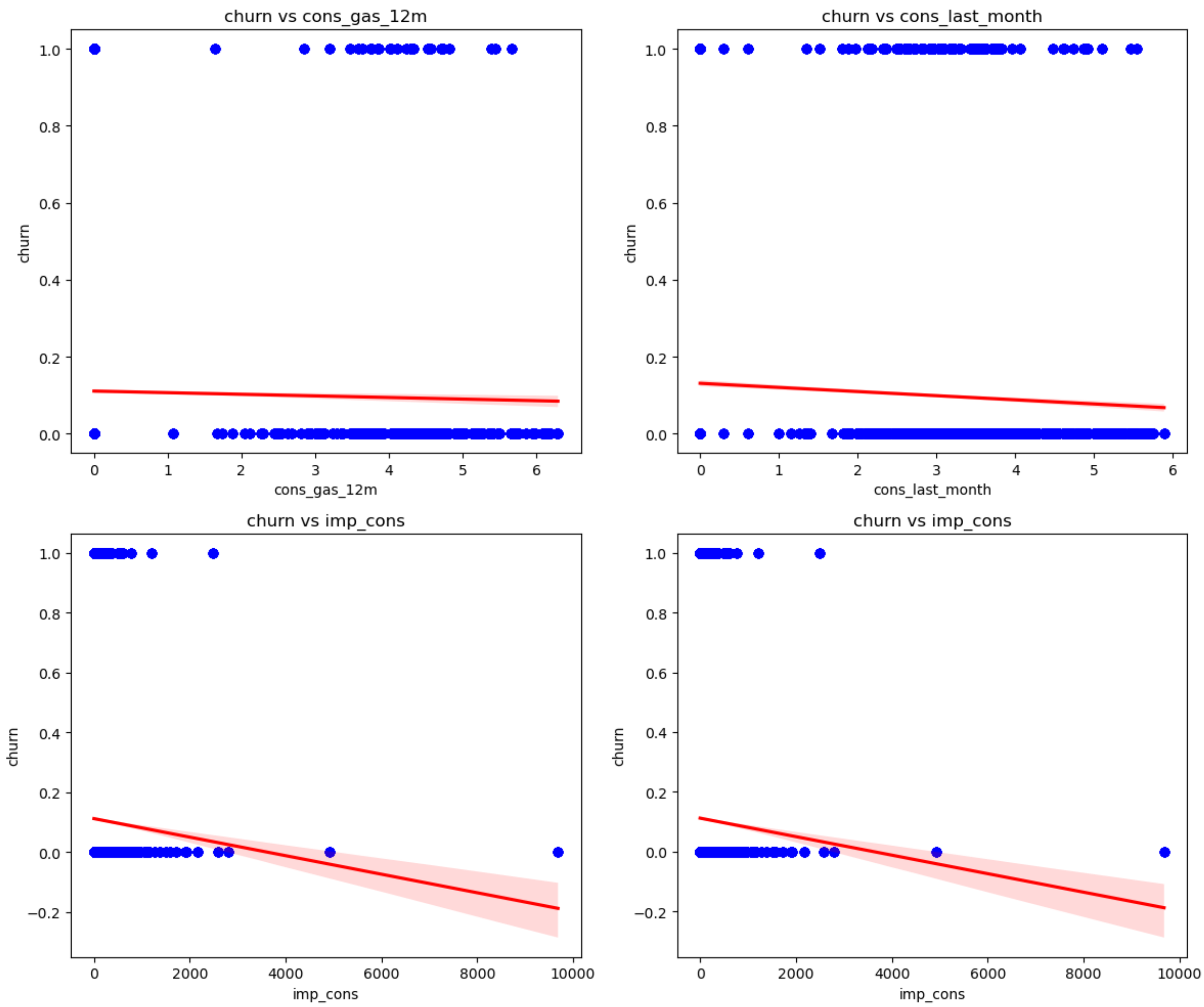
Out[42]:

	id	cons_12m	cons_gas_12m	coı
0	24011ae4ebbe3035111d65fa7c15bc57	0.000000	4.739944	
1	24011ae4ebbe3035111d65fa7c15bc57	0.000000	4.739944	
2	24011ae4ebbe3035111d65fa7c15bc57	0.000000	4.739944	
3	24011ae4ebbe3035111d65fa7c15bc57	0.000000	4.739944	
4	24011ae4ebbe3035111d65fa7c15bc57	0.000000	4.739944	
...	...	...	...	
16091	0082e565c1298cbe9cf70e96f571e4fa	3.990339	2.546543	
16092	e84c8b7d0e7e31cfebb07f46ac7445cf	3.861773	5.270579	
16093	e84c8b7d0e7e31cfebb07f46ac7445cf	3.861773	5.270579	
16094	e84c8b7d0e7e31cfebb07f46ac7445cf	3.861773	5.270579	
16095	e84c8b7d0e7e31cfebb07f46ac7445cf	3.861773	5.270579	

16096 rows × 42 columns

In [4…

```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(12, 10))

# Regression plot for 'cons_12m' vs 'cons_gas_12m'
sns.regplot(y='churn', x='cons_gas_12m', data=df,scatter_kws=
axs[0, 0].set_title('churn vs cons_gas_12m')

# Regression plot for 'cons_12m' vs 'cons_last_month'
sns.regplot(y='churn', x='cons_last_month', data=df,scatter_k
axs[0, 1].set_title('churn vs cons_last_month')

# Regression plot for 'cons_12m' vs 'imp_cons'
sns.regplot(y='churn', x='imp_cons', data=df,scatter_kws={'cc
axs[1, 0].set_title('churn vs imp_cons')

# Regression plot for 'cons_gas_12m' vs 'imp_cons'
sns.regplot(y='churn', x='imp_cons', data=df,scatter_kws={'cc
axs[1, 1].set_title('churn vs imp_cons')


plt.tight_layout()

plt.show()
```



In [4…

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_re
```

In [4…

```
Y = df['churn']
X = df.drop(columns=['id', 'churn'])
X_train, X_tests, Y_train, Y_tests = train_test_split(X, Y, t
print ('Train set:', X_train.shape,  Y_train.shape)
print ('Test set:', X_tests.shape,  Y_tests.shape)
```

```
Train set: (11267, 40) (11267,)
Test set: (4829, 40) (4829,)
```

In [4…
```
def model_fit_predict(model, X, Y, X_predict):
    model.fit(X,Y)
    return model.predict(X_predict)
def acc_score(Y_true, Y_pred):
    return accuracy_score(Y_true, Y_pred)
def pre_score(Y_true,Y_pred):
    return precision_score(Y_true, Y_pred)
def f_score(Y_true, Y_pred):
    return f1_score(Y_true, Y_pred)
```

In [4…
```
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn import svm
```

In […
```

```

In [4…
```
model1 = RandomForestClassifier(n_estimators=1000)

Y_pred_test = model_fit_predict(model1, X_train, Y_train, X_t

#f1 score for training data
f1 = round(f1_score(Y_tests, Y_pred_test),2)

#accuracy score for training data
acc = round(accuracy_score(Y_tests, Y_pred_test),2)

#precision score for training data
pre = round(precision_score(Y_tests, Y_pred_test),2)

print(f"Accuracy, precision and f1-score for training data ar
```

```
Accuracy, precision and f1-score for training data are 1.0,
1.0 and 1.0 respectively
```

In [4…

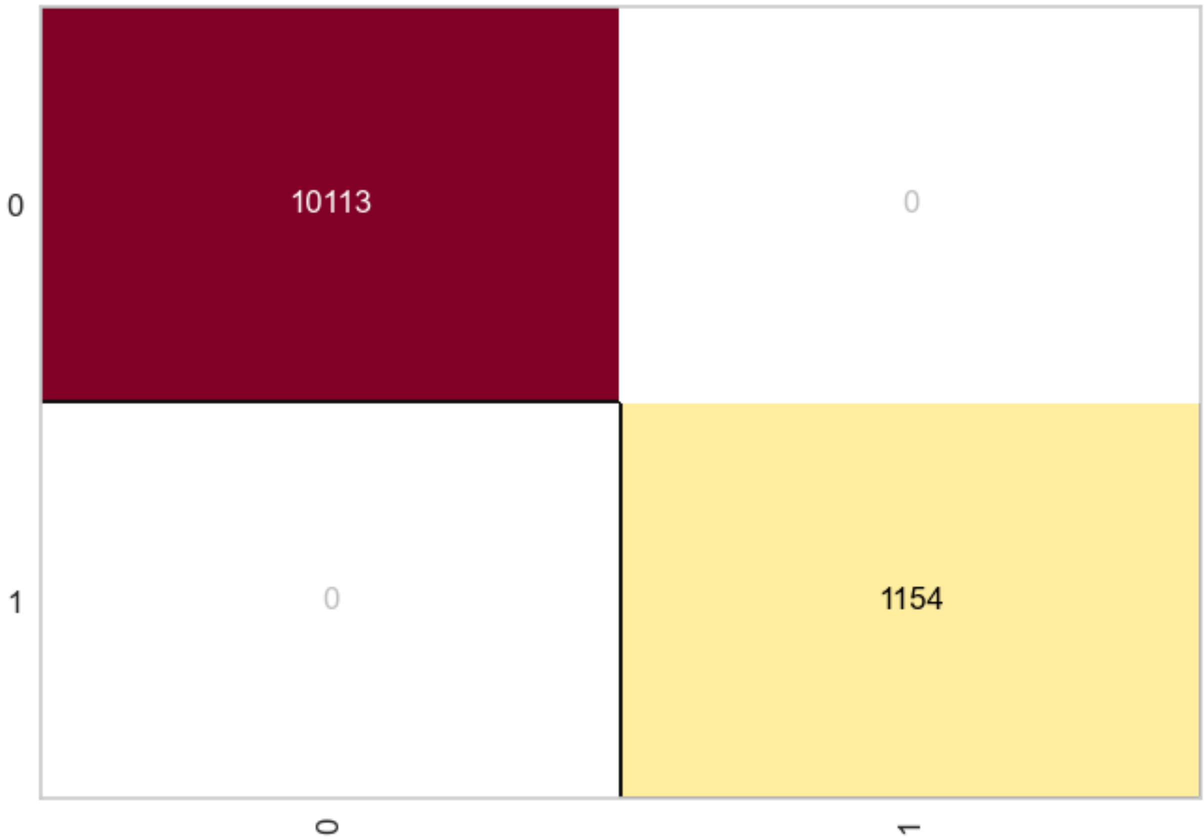
```
from sklearn import metrics
predictions1 = model1.predict(X_tests)
tn, fp, fn, tp = metrics.confusion_matrix(Y_tests, predictior
Y_tests.value_counts()


print(f"True positives: {tp}")
print(f"False positives: {fp}")
print(f"True negatives: {tn}")
print(f"False negatives: {fn}\n")

print(f"Accuracy: {metrics.accuracy_score(Y_tests, predictior
print(f"Precision: {metrics.precision_score(Y_tests, predicti
print(f"Recall: {metrics.recall_score(Y_tests, predictions1)}
```

```
True positives: 572
False positives: 0
True negatives: 4257
False negatives: 0

Accuracy: 1.0
Precision: 1.0
Recall: 1.0
```

In [5…

```
from sklearn.metrics import confusion_matrix
from yellowbrick.classifier import ConfusionMatrix
cm1= ConfusionMatrix(model1, classes=[0,1])
cm1.fit(X_train, Y_train)

cm1.score(X_train, Y_train)
```

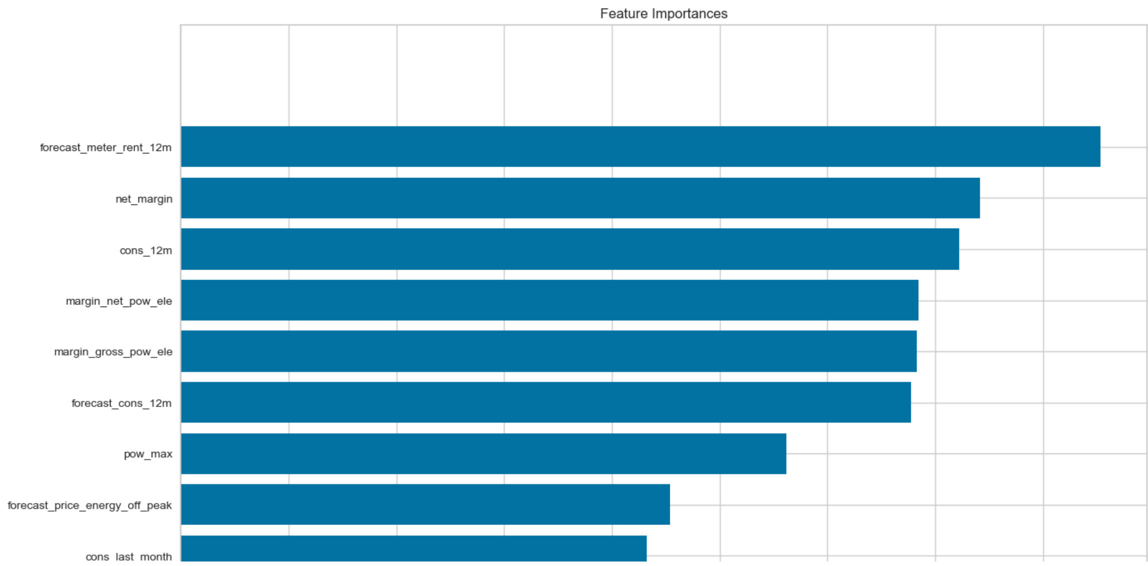
Out[50]: 1.0

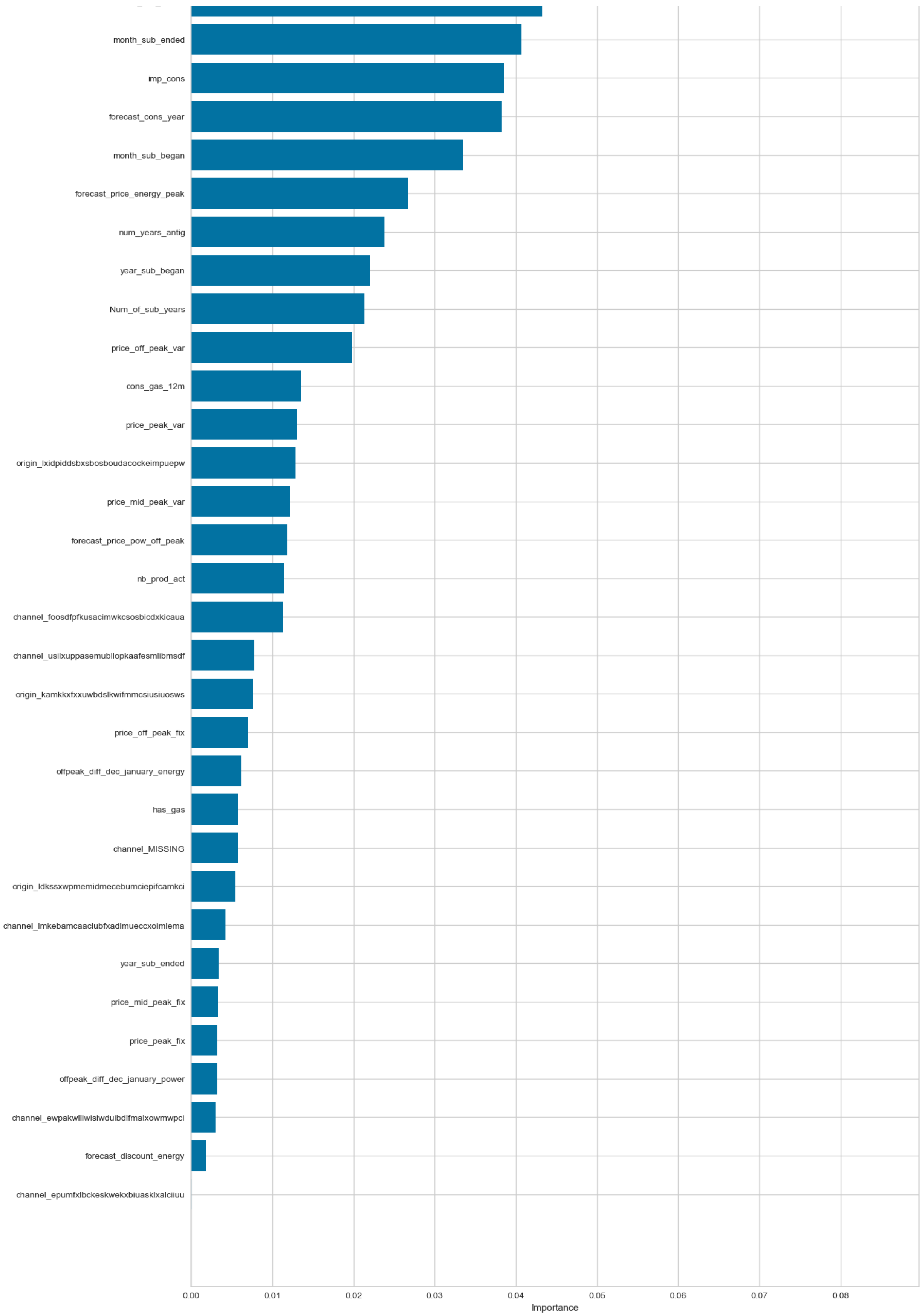


Evaluating Features to fix Overfitting

In [5…
```
feature_importances = pd.DataFrame({'features': X_train.colum
```

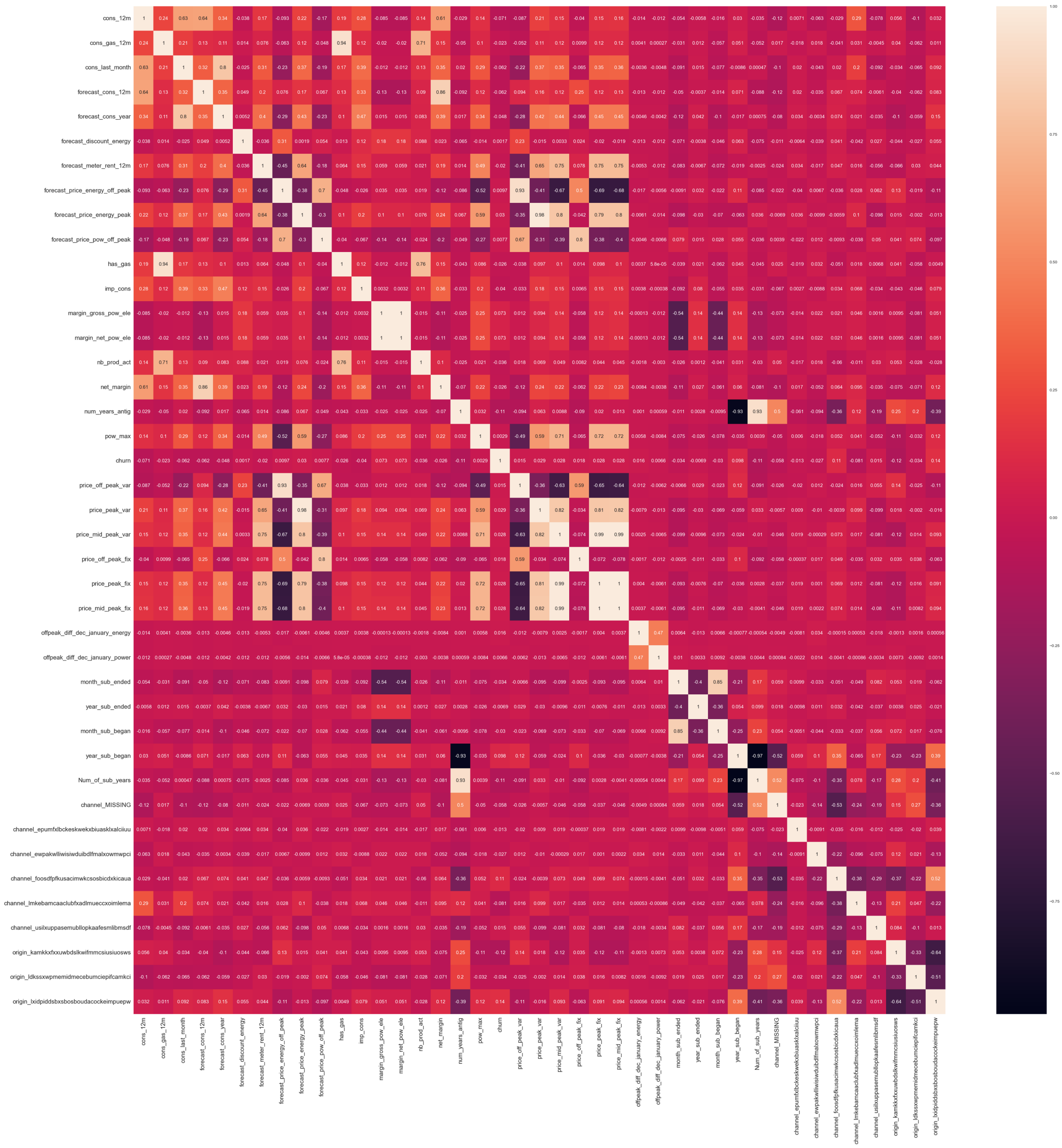
In [5…
```
plt.figure(figsize=(15, 35))
plt.title('Feature Importances')
plt.barh(range(len(feature_importances)), feature_importances
plt.yticks(range(len(feature_importances)), feature_importanc
plt.xlabel('Importance')
plt.show()
```




month_sub_ended
imp_cons
forecast_cons_year
month_sub_began
forecast_price_energy_peak
num_years_antig
year_sub_began
Num_of_sub_years
price_off_peak_var
cons_gas_12m
price_peak_var
origin_lxidpiddsbxsbosboudacockeimpuepw
price_mid_peak_var
forecast_price_pow_off_peak
nb_prod_act
channel_foosdfpfkusacimwkcsosbicdxkicaua
channel_usilxuppasemubllopkaafesmlibmsdf
origin_kamkkxfxxuwbdslkwifmmcsiusiuosws
price_off_peak_fix
offpeak_diff_dec_january_energy
has_gas
channel_MISSING
origin_ldkssxwpmemidmecebumciepifcamkci
channel_lmkebamcaaclubfxadlmueccxoimlema
year_sub_ended
price_mid_peak_fix
price_peak_fix
offpeak_diff_dec_january_power
channel_ewpakwlliwisiwduibdlfmalxowmwpci
forecast_discount_energy
channel_epumfxlbckeskwekxbiuasklxalciiuu
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Importance

In [5…

```
correlation = df.drop(columns=['id']).corr()
plt.figure(figsize=(45, 45))
sns.heatmap(
correlation, xticklabels=correlation.columns.values, yticklab
annot_kws={'size': 12}
)
# Axis ticks size
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.show()
```



In [5…

```
def correlation (dataset, threshold):
    col_corr = set() # Set of all the names of correlated col
    corr_matrix = dataset.corr()
    for i in range (len (corr_matrix.columns)) :
        for j in range(i):
            if abs (corr_matrix.iloc [i, j]) > threshold: # w
                colname = corr_matrix.columns [i] # getting t
                col_corr.add(colname)
    return col_corr
```

In [5…

```
corr_features=correlation(X_train, 0.9)
len(set(corr_features))
```

Out[55]:

```
8
```

In [5…

```
corr_features
```

Out[56]:

```
{'Num_of_sub_years',
 'has_gas',
 'margin_net_pow_ele',
 'price_mid_peak_fix',
 'price_off_peak_var',
 'price_peak_fix',
 'price_peak_var',
 'year_sub_began'}
```

In […

5.Further Feature Selection

In [5…

```
features_to_drop=['Num_of_sub_years','has_gas','margin_net_pc
model_df=df.drop(columns=features_to_drop,axis=1)
model_df.head(5)
```

Out[57]:

	id	cons_12m	cons_gas_12m	cons_las
0	24011ae4ebbe3035111d65fa7c15bc57	0.0	4.739944	
1	24011ae4ebbe3035111d65fa7c15bc57	0.0	4.739944	
2	24011ae4ebbe3035111d65fa7c15bc57	0.0	4.739944	
3	24011ae4ebbe3035111d65fa7c15bc57	0.0	4.739944	
4	24011ae4ebbe3035111d65fa7c15bc57	0.0	4.739944	

5 rows × 37 columns

In […

In [5…

```
Y = model_df['churn']
X = model_df.drop(columns=['id', 'churn'])
X_train, X_tests, Y_train, Y_tests = train_test_split(X, Y, t
print ('Train set:', X_train.shape,  Y_train.shape)
print ('Test set:', X_tests.shape,  Y_tests.shape)
```

```
Train set: (12072, 35) (12072,)
Test set: (4024, 35) (4024,)
```

In […

In [5…

```
model2 = RandomForestClassifier(n_estimators=1000)


Y_pred_test = model_fit_predict(model2, X_train, Y_train, X_t

#f1 score for training data
f1 = round(f1_score(Y_tests, Y_pred_test),2)

#accuracy score for training data
acc = round(accuracy_score(Y_tests, Y_pred_test),2)

#precision score for training data
pre = round(precision_score(Y_tests, Y_pred_test),2)

print(f"Accuracy, precision and f1-score for training data ar
```

```
Accuracy, precision and f1-score for training data are 1.0,
1.0 and 1.0 respectively
```

In [6…

```
predictions2 = model2.predict(X_tests)
tn, fp, fn, tp = metrics.confusion_matrix(Y_tests, predictior
Y_tests.value_counts()
```

Out[60]:

```
churn
0    3556
1     468
Name: count, dtype: int64
```

In [6…

```
print("Shape of Y_tests:", Y_tests.shape)
print("Shape of predictions:", predictions2.shape)
print("Y_tests values:", Y_tests)
print("Predictions values:", predictions2)
```

```
Shape of Y_tests: (4024,)
Shape of predictions: (4024,)
Y_tests values: 4679     0
102      0
11366    0
2816     0
9317     0
        ..
2470     0
12465    1
15347    0
9701     0
11924    0
Name: churn, Length: 4024, dtype: int64
Predictions values: [0 0 0 ... 0 0 0]
```

In [6…

```
from sklearn import ensemble, model_selection

classifier = ensemble.RandomForestClassifier(n_jobs=-1)
param_grid= {
    'n_estimators':[100,500,1000],
    'max_depth':[1,3,5,7],
    'criterion':["gini","entropy"],

}

model=model_selection.GridSearchCV(
    estimator=classifier,
    param_grid=param_grid,
    scoring="accuracy",
    verbose=10,
    n_jobs=1,
    cv=5,
)

model.fit(X,Y)
print(model.best_score_)
print(model.best_estimator_.get_params())
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fit
s
[CV 1/5; 1/24] START criterion=gini, max_depth=1, n_estimato
rs=100..............
[CV 1/5; 1/24] END criterion=gini, max_depth=1, n_estimators
=100;, score=0.893 total time=   3.3s
[CV 2/5; 1/24] START criterion=gini, max_depth=1, n_estimato
rs=100..............
[CV 2/5; 1/24] END criterion=gini, max_depth=1, n_estimators
=100;, score=0.893 total time=   0.3s
[CV 3/5; 1/24] START criterion=gini, max_depth=1, n_estimato
rs=100..............
[CV 3/5; 1/24] END criterion=gini, max_depth=1, n_estimators
=100;, score=0.893 total time=   0.3s
[CV 4/5; 1/24] START criterion=gini, max_depth=1, n_estimato
rs=100..............
[CV 4/5; 1/24] END criterion=gini, max_depth=1, n_estimators
=100;, score=0.893 total time=   0.4s
[CV 5/5; 1/24] START criterion=gini, max_depth=1, n_estimato
rs=100..............
[CV 5/5; 1/24] END criterion=gini, max_depth=1, n_estimators
=100;, score=0.893 total time=   0.3s
[CV 1/5; 2/24] START criterion=gini, max_depth=1, n_estimato
rs=500..............
```

```
[CV 1/5; 2/24] END criterion=gini, max_depth=1, n_estimators
=500;, score=0.893 total time=   1.0s
[CV 2/5; 2/24] START criterion=gini, max_depth=1, n_estimato
rs=500.............
[CV 2/5; 2/24] END criterion=gini, max_depth=1, n_estimators
=500;, score=0.893 total time=   1.0s
[CV 3/5; 2/24] START criterion=gini, max_depth=1, n_estimato
rs=500.............
[CV 3/5; 2/24] END criterion=gini, max_depth=1, n_estimators
=500;, score=0.893 total time=   1.1s
[CV 4/5; 2/24] START criterion=gini, max_depth=1, n_estimato
rs=500.............
[CV 4/5; 2/24] END criterion=gini, max_depth=1, n_estimators
=500;, score=0.893 total time=   1.7s
[CV 5/5; 2/24] START criterion=gini, max_depth=1, n_estimato
rs=500.............
[CV 5/5; 2/24] END criterion=gini, max_depth=1, n_estimators
=500;, score=0.893 total time=   1.6s
[CV 1/5; 3/24] START criterion=gini, max_depth=1, n_estimato
rs=1000............
[CV 1/5; 3/24] END criterion=gini, max_depth=1, n_estimators
=1000;, score=0.893 total time=   2.1s
[CV 2/5; 3/24] START criterion=gini, max_depth=1, n_estimato
rs=1000............
[CV 2/5; 3/24] END criterion=gini, max_depth=1, n_estimators
=1000;, score=0.893 total time=   2.2s
[CV 3/5; 3/24] START criterion=gini, max_depth=1, n_estimato
rs=1000............
[CV 3/5; 3/24] END criterion=gini, max_depth=1, n_estimators
=1000;, score=0.893 total time=   2.1s
[CV 4/5; 3/24] START criterion=gini, max_depth=1, n_estimato
rs=1000............
[CV 4/5; 3/24] END criterion=gini, max_depth=1, n_estimators
=1000;, score=0.893 total time=   2.5s
[CV 5/5; 3/24] START criterion=gini, max_depth=1, n_estimato
rs=1000............
[CV 5/5; 3/24] END criterion=gini, max_depth=1, n_estimators
=1000;, score=0.893 total time=   2.4s
[CV 1/5; 4/24] START criterion=gini, max_depth=3, n_estimato
rs=100.............
[CV 1/5; 4/24] END criterion=gini, max_depth=3, n_estimators
=100;, score=0.893 total time=   0.4s
[CV 2/5; 4/24] START criterion=gini, max_depth=3, n_estimato
rs=100.............
[CV 2/5; 4/24] END criterion=gini, max_depth=3, n_estimators
=100;, score=0.893 total time=   0.5s
[CV 3/5; 4/24] START criterion=gini, max_depth=3, n_estimato
rs=100.............
```

```
[CV 3/5; 4/24] END criterion=gini, max_depth=3, n_estimators
=100;, score=0.893 total time=   0.5s
[CV 4/5; 4/24] START criterion=gini, max_depth=3, n_estimato
rs=100.............
[CV 4/5; 4/24] END criterion=gini, max_depth=3, n_estimators
=100;, score=0.893 total time=   0.5s
[CV 5/5; 4/24] START criterion=gini, max_depth=3, n_estimato
rs=100.............
[CV 5/5; 4/24] END criterion=gini, max_depth=3, n_estimators
=100;, score=0.893 total time=   0.5s
[CV 1/5; 5/24] START criterion=gini, max_depth=3, n_estimato
rs=500.............
[CV 1/5; 5/24] END criterion=gini, max_depth=3, n_estimators
=500;, score=0.893 total time=   1.6s
[CV 2/5; 5/24] START criterion=gini, max_depth=3, n_estimato
rs=500.............
[CV 2/5; 5/24] END criterion=gini, max_depth=3, n_estimators
=500;, score=0.893 total time=   1.5s
[CV 3/5; 5/24] START criterion=gini, max_depth=3, n_estimato
rs=500.............
[CV 3/5; 5/24] END criterion=gini, max_depth=3, n_estimators
=500;, score=0.893 total time=   1.6s
[CV 4/5; 5/24] START criterion=gini, max_depth=3, n_estimato
rs=500.............
[CV 4/5; 5/24] END criterion=gini, max_depth=3, n_estimators
=500;, score=0.893 total time=   1.6s
[CV 5/5; 5/24] START criterion=gini, max_depth=3, n_estimato
rs=500.............
[CV 5/5; 5/24] END criterion=gini, max_depth=3, n_estimators
=500;, score=0.893 total time=   1.6s
[CV 1/5; 6/24] START criterion=gini, max_depth=3, n_estimato
rs=1000............
[CV 1/5; 6/24] END criterion=gini, max_depth=3, n_estimators
=1000;, score=0.893 total time=   3.9s
[CV 2/5; 6/24] START criterion=gini, max_depth=3, n_estimato
rs=1000............
[CV 2/5; 6/24] END criterion=gini, max_depth=3, n_estimators
=1000;, score=0.893 total time=   3.2s
[CV 3/5; 6/24] START criterion=gini, max_depth=3, n_estimato
rs=1000............
[CV 3/5; 6/24] END criterion=gini, max_depth=3, n_estimators
=1000;, score=0.893 total time=   3.3s
[CV 4/5; 6/24] START criterion=gini, max_depth=3, n_estimato
rs=1000............
[CV 4/5; 6/24] END criterion=gini, max_depth=3, n_estimators
=1000;, score=0.893 total time=   3.0s
[CV 5/5; 6/24] START criterion=gini, max_depth=3, n_estimato
rs=1000............
```

```
[CV 5/5; 6/24] END criterion=gini, max_depth=3, n_estimators
=1000;, score=0.893 total time=   2.9s
[CV 1/5; 7/24] START criterion=gini, max_depth=5, n_estimato
rs=100.............
[CV 1/5; 7/24] END criterion=gini, max_depth=5, n_estimators
=100;, score=0.893 total time=   0.6s
[CV 2/5; 7/24] START criterion=gini, max_depth=5, n_estimato
rs=100.............
[CV 2/5; 7/24] END criterion=gini, max_depth=5, n_estimators
=100;, score=0.889 total time=   0.6s
[CV 3/5; 7/24] START criterion=gini, max_depth=5, n_estimato
rs=100.............
[CV 3/5; 7/24] END criterion=gini, max_depth=5, n_estimators
=100;, score=0.893 total time=   0.5s
[CV 4/5; 7/24] START criterion=gini, max_depth=5, n_estimato
rs=100.............
[CV 4/5; 7/24] END criterion=gini, max_depth=5, n_estimators
=100;, score=0.893 total time=   0.6s
[CV 5/5; 7/24] START criterion=gini, max_depth=5, n_estimato
rs=100.............
[CV 5/5; 7/24] END criterion=gini, max_depth=5, n_estimators
=100;, score=0.893 total time=   0.5s
[CV 1/5; 8/24] START criterion=gini, max_depth=5, n_estimato
rs=500.............
[CV 1/5; 8/24] END criterion=gini, max_depth=5, n_estimators
=500;, score=0.893 total time=   1.9s
[CV 2/5; 8/24] START criterion=gini, max_depth=5, n_estimato
rs=500.............
[CV 2/5; 8/24] END criterion=gini, max_depth=5, n_estimators
=500;, score=0.889 total time=   2.6s
[CV 3/5; 8/24] START criterion=gini, max_depth=5, n_estimato
rs=500.............
[CV 3/5; 8/24] END criterion=gini, max_depth=5, n_estimators
=500;, score=0.893 total time=   2.0s
[CV 4/5; 8/24] START criterion=gini, max_depth=5, n_estimato
rs=500.............
[CV 4/5; 8/24] END criterion=gini, max_depth=5, n_estimators
=500;, score=0.893 total time=   2.6s
[CV 5/5; 8/24] START criterion=gini, max_depth=5, n_estimato
rs=500.............
[CV 5/5; 8/24] END criterion=gini, max_depth=5, n_estimators
=500;, score=0.893 total time=   2.3s
[CV 1/5; 9/24] START criterion=gini, max_depth=5, n_estimato
rs=1000............
[CV 1/5; 9/24] END criterion=gini, max_depth=5, n_estimators
=1000;, score=0.893 total time=   4.5s
[CV 2/5; 9/24] START criterion=gini, max_depth=5, n_estimato
rs=1000............
```

```
[CV 2/5; 9/24] END criterion=gini, max_depth=5, n_estimators
=1000;, score=0.889 total time=   4.4s
[CV 3/5; 9/24] START criterion=gini, max_depth=5, n_estimato
rs=1000............
[CV 3/5; 9/24] END criterion=gini, max_depth=5, n_estimators
=1000;, score=0.893 total time=   4.0s
[CV 4/5; 9/24] START criterion=gini, max_depth=5, n_estimato
rs=1000............
[CV 4/5; 9/24] END criterion=gini, max_depth=5, n_estimators
=1000;, score=0.893 total time=   3.7s
[CV 5/5; 9/24] START criterion=gini, max_depth=5, n_estimato
rs=1000............
[CV 5/5; 9/24] END criterion=gini, max_depth=5, n_estimators
=1000;, score=0.893 total time=   3.8s
[CV 1/5; 10/24] START criterion=gini, max_depth=7, n_estimat
ors=100............
[CV 1/5; 10/24] END criterion=gini, max_depth=7, n_estimator
s=100;, score=0.893 total time=   0.6s
[CV 2/5; 10/24] START criterion=gini, max_depth=7, n_estimat
ors=100............
[CV 2/5; 10/24] END criterion=gini, max_depth=7, n_estimator
s=100;, score=0.883 total time=   0.8s
[CV 3/5; 10/24] START criterion=gini, max_depth=7, n_estimat
ors=100............
[CV 3/5; 10/24] END criterion=gini, max_depth=7, n_estimator
s=100;, score=0.893 total time=   0.9s
[CV 4/5; 10/24] START criterion=gini, max_depth=7, n_estimat
ors=100............
[CV 4/5; 10/24] END criterion=gini, max_depth=7, n_estimator
s=100;, score=0.894 total time=   0.6s
[CV 5/5; 10/24] START criterion=gini, max_depth=7, n_estimat
ors=100............
[CV 5/5; 10/24] END criterion=gini, max_depth=7, n_estimator
s=100;, score=0.893 total time=   0.7s
[CV 1/5; 11/24] START criterion=gini, max_depth=7, n_estimat
ors=500............
[CV 1/5; 11/24] END criterion=gini, max_depth=7, n_estimator
s=500;, score=0.893 total time=   2.4s
[CV 2/5; 11/24] START criterion=gini, max_depth=7, n_estimat
ors=500............
[CV 2/5; 11/24] END criterion=gini, max_depth=7, n_estimator
s=500;, score=0.884 total time=   2.4s
[CV 3/5; 11/24] START criterion=gini, max_depth=7, n_estimat
ors=500............
[CV 3/5; 11/24] END criterion=gini, max_depth=7, n_estimator
s=500;, score=0.893 total time=   2.3s
[CV 4/5; 11/24] START criterion=gini, max_depth=7, n_estimat
ors=500............
```

```
[CV 4/5; 11/24] END criterion=gini, max_depth=7, n_estimator
s=500;, score=0.897 total time=   2.4s
[CV 5/5; 11/24] START criterion=gini, max_depth=7, n_estimat
ors=500.............
[CV 5/5; 11/24] END criterion=gini, max_depth=7, n_estimator
s=500;, score=0.893 total time=   2.7s
[CV 1/5; 12/24] START criterion=gini, max_depth=7, n_estimat
ors=1000............
[CV 1/5; 12/24] END criterion=gini, max_depth=7, n_estimator
s=1000;, score=0.893 total time=   6.2s
[CV 2/5; 12/24] START criterion=gini, max_depth=7, n_estimat
ors=1000............
[CV 2/5; 12/24] END criterion=gini, max_depth=7, n_estimator
s=1000;, score=0.884 total time=   4.6s
[CV 3/5; 12/24] START criterion=gini, max_depth=7, n_estimat
ors=1000............
[CV 3/5; 12/24] END criterion=gini, max_depth=7, n_estimator
s=1000;, score=0.893 total time=   4.5s
[CV 4/5; 12/24] START criterion=gini, max_depth=7, n_estimat
ors=1000............
[CV 4/5; 12/24] END criterion=gini, max_depth=7, n_estimator
s=1000;, score=0.897 total time=   4.9s
[CV 5/5; 12/24] START criterion=gini, max_depth=7, n_estimat
ors=1000............
[CV 5/5; 12/24] END criterion=gini, max_depth=7, n_estimator
s=1000;, score=0.897 total time=   4.9s
[CV 1/5; 13/24] START criterion=entropy, max_depth=1, n_esti
mators=100..........
[CV 1/5; 13/24] END criterion=entropy, max_depth=1, n_estima
tors=100;, score=0.893 total time=   0.4s
[CV 2/5; 13/24] START criterion=entropy, max_depth=1, n_esti
mators=100..........
[CV 2/5; 13/24] END criterion=entropy, max_depth=1, n_estima
tors=100;, score=0.893 total time=   0.4s
[CV 3/5; 13/24] START criterion=entropy, max_depth=1, n_esti
mators=100..........
[CV 3/5; 13/24] END criterion=entropy, max_depth=1, n_estima
tors=100;, score=0.893 total time=   0.4s
[CV 4/5; 13/24] START criterion=entropy, max_depth=1, n_esti
mators=100..........
[CV 4/5; 13/24] END criterion=entropy, max_depth=1, n_estima
tors=100;, score=0.893 total time=   0.5s
[CV 5/5; 13/24] START criterion=entropy, max_depth=1, n_esti
mators=100..........
[CV 5/5; 13/24] END criterion=entropy, max_depth=1, n_estima
tors=100;, score=0.893 total time=   0.3s
[CV 1/5; 14/24] START criterion=entropy, max_depth=1, n_esti
mators=500..........
```

```
[CV 1/5; 14/24] END criterion=entropy, max_depth=1, n_estima
tors=500;, score=0.893 total time=   1.1s
[CV 2/5; 14/24] START criterion=entropy, max_depth=1, n_esti
mators=500..........
[CV 2/5; 14/24] END criterion=entropy, max_depth=1, n_estima
tors=500;, score=0.893 total time=   1.1s
[CV 3/5; 14/24] START criterion=entropy, max_depth=1, n_esti
mators=500..........
[CV 3/5; 14/24] END criterion=entropy, max_depth=1, n_estima
tors=500;, score=0.893 total time=   1.1s
[CV 4/5; 14/24] START criterion=entropy, max_depth=1, n_esti
mators=500..........
[CV 4/5; 14/24] END criterion=entropy, max_depth=1, n_estima
tors=500;, score=0.893 total time=   1.2s
[CV 5/5; 14/24] START criterion=entropy, max_depth=1, n_esti
mators=500..........
[CV 5/5; 14/24] END criterion=entropy, max_depth=1, n_estima
tors=500;, score=0.893 total time=   1.1s
[CV 1/5; 15/24] START criterion=entropy, max_depth=1, n_esti
mators=1000.........
[CV 1/5; 15/24] END criterion=entropy, max_depth=1, n_estima
tors=1000;, score=0.893 total time=   2.1s
[CV 2/5; 15/24] START criterion=entropy, max_depth=1, n_esti
mators=1000.........
[CV 2/5; 15/24] END criterion=entropy, max_depth=1, n_estima
tors=1000;, score=0.893 total time=   2.3s
[CV 3/5; 15/24] START criterion=entropy, max_depth=1, n_esti
mators=1000.........
[CV 3/5; 15/24] END criterion=entropy, max_depth=1, n_estima
tors=1000;, score=0.893 total time=   2.1s
[CV 4/5; 15/24] START criterion=entropy, max_depth=1, n_esti
mators=1000.........
[CV 4/5; 15/24] END criterion=entropy, max_depth=1, n_estima
tors=1000;, score=0.893 total time=   2.6s
[CV 5/5; 15/24] START criterion=entropy, max_depth=1, n_esti
mators=1000.........
[CV 5/5; 15/24] END criterion=entropy, max_depth=1, n_estima
tors=1000;, score=0.893 total time=   2.3s
[CV 1/5; 16/24] START criterion=entropy, max_depth=3, n_esti
mators=100..........
[CV 1/5; 16/24] END criterion=entropy, max_depth=3, n_estima
tors=100;, score=0.893 total time=   0.5s
[CV 2/5; 16/24] START criterion=entropy, max_depth=3, n_esti
mators=100..........
[CV 2/5; 16/24] END criterion=entropy, max_depth=3, n_estima
tors=100;, score=0.893 total time=   0.5s
[CV 3/5; 16/24] START criterion=entropy, max_depth=3, n_esti
mators=100..........
```

```
[CV 3/5; 16/24] END criterion=entropy, max_depth=3, n_estima
tors=100;, score=0.893 total time=   0.6s
[CV 4/5; 16/24] START criterion=entropy, max_depth=3, n_esti
mators=100..........
[CV 4/5; 16/24] END criterion=entropy, max_depth=3, n_estima
tors=100;, score=0.893 total time=   0.5s
[CV 5/5; 16/24] START criterion=entropy, max_depth=3, n_esti
mators=100..........
[CV 5/5; 16/24] END criterion=entropy, max_depth=3, n_estima
tors=100;, score=0.893 total time=   0.4s
[CV 1/5; 17/24] START criterion=entropy, max_depth=3, n_esti
mators=500..........
[CV 1/5; 17/24] END criterion=entropy, max_depth=3, n_estima
tors=500;, score=0.893 total time=   1.7s
[CV 2/5; 17/24] START criterion=entropy, max_depth=3, n_esti
mators=500..........
[CV 2/5; 17/24] END criterion=entropy, max_depth=3, n_estima
tors=500;, score=0.893 total time=   1.6s
[CV 3/5; 17/24] START criterion=entropy, max_depth=3, n_esti
mators=500..........
[CV 3/5; 17/24] END criterion=entropy, max_depth=3, n_estima
tors=500;, score=0.893 total time=   1.7s
[CV 4/5; 17/24] START criterion=entropy, max_depth=3, n_esti
mators=500..........
[CV 4/5; 17/24] END criterion=entropy, max_depth=3, n_estima
tors=500;, score=0.893 total time=   1.6s
[CV 5/5; 17/24] START criterion=entropy, max_depth=3, n_esti
mators=500..........
[CV 5/5; 17/24] END criterion=entropy, max_depth=3, n_estima
tors=500;, score=0.893 total time=   1.6s
[CV 1/5; 18/24] START criterion=entropy, max_depth=3, n_esti
mators=1000.........
[CV 1/5; 18/24] END criterion=entropy, max_depth=3, n_estima
tors=1000;, score=0.893 total time=   3.6s
[CV 2/5; 18/24] START criterion=entropy, max_depth=3, n_esti
mators=1000.........
[CV 2/5; 18/24] END criterion=entropy, max_depth=3, n_estima
tors=1000;, score=0.893 total time=   3.2s
[CV 3/5; 18/24] START criterion=entropy, max_depth=3, n_esti
mators=1000.........
[CV 3/5; 18/24] END criterion=entropy, max_depth=3, n_estima
tors=1000;, score=0.893 total time=   3.0s
[CV 4/5; 18/24] START criterion=entropy, max_depth=3, n_esti
mators=1000.........
[CV 4/5; 18/24] END criterion=entropy, max_depth=3, n_estima
tors=1000;, score=0.893 total time=   3.1s
[CV 5/5; 18/24] START criterion=entropy, max_depth=3, n_esti
mators=1000.........
```

```
[CV 5/5; 18/24] END criterion=entropy, max_depth=3, n_estima
tors=1000;, score=0.893 total time=   3.3s
[CV 1/5; 19/24] START criterion=entropy, max_depth=5, n_esti
mators=100..........
[CV 1/5; 19/24] END criterion=entropy, max_depth=5, n_estima
tors=100;, score=0.893 total time=   0.6s
[CV 2/5; 19/24] START criterion=entropy, max_depth=5, n_esti
mators=100..........
[CV 2/5; 19/24] END criterion=entropy, max_depth=5, n_estima
tors=100;, score=0.893 total time=   0.7s
[CV 3/5; 19/24] START criterion=entropy, max_depth=5, n_esti
mators=100..........
[CV 3/5; 19/24] END criterion=entropy, max_depth=5, n_estima
tors=100;, score=0.893 total time=   0.6s
[CV 4/5; 19/24] START criterion=entropy, max_depth=5, n_esti
mators=100..........
[CV 4/5; 19/24] END criterion=entropy, max_depth=5, n_estima
tors=100;, score=0.893 total time=   0.6s
[CV 5/5; 19/24] START criterion=entropy, max_depth=5, n_esti
mators=100..........
[CV 5/5; 19/24] END criterion=entropy, max_depth=5, n_estima
tors=100;, score=0.893 total time=   0.6s
[CV 1/5; 20/24] START criterion=entropy, max_depth=5, n_esti
mators=500..........
[CV 1/5; 20/24] END criterion=entropy, max_depth=5, n_estima
tors=500;, score=0.893 total time=   2.3s
[CV 2/5; 20/24] START criterion=entropy, max_depth=5, n_esti
mators=500..........
[CV 2/5; 20/24] END criterion=entropy, max_depth=5, n_estima
tors=500;, score=0.889 total time=   2.3s
[CV 3/5; 20/24] START criterion=entropy, max_depth=5, n_esti
mators=500..........
[CV 3/5; 20/24] END criterion=entropy, max_depth=5, n_estima
tors=500;, score=0.893 total time=   2.1s
[CV 4/5; 20/24] START criterion=entropy, max_depth=5, n_esti
mators=500..........
[CV 4/5; 20/24] END criterion=entropy, max_depth=5, n_estima
tors=500;, score=0.893 total time=   2.2s
[CV 5/5; 20/24] START criterion=entropy, max_depth=5, n_esti
mators=500..........
[CV 5/5; 20/24] END criterion=entropy, max_depth=5, n_estima
tors=500;, score=0.893 total time=   2.4s
[CV 1/5; 21/24] START criterion=entropy, max_depth=5, n_esti
mators=1000.........
[CV 1/5; 21/24] END criterion=entropy, max_depth=5, n_estima
tors=1000;, score=0.893 total time=   4.1s
[CV 2/5; 21/24] START criterion=entropy, max_depth=5, n_esti
mators=1000.........
```

```
[CV 2/5; 21/24] END criterion=entropy, max_depth=5, n_estima
tors=1000;, score=0.893 total time=   4.1s
[CV 3/5; 21/24] START criterion=entropy, max_depth=5, n_esti
mators=1000.........
[CV 3/5; 21/24] END criterion=entropy, max_depth=5, n_estima
tors=1000;, score=0.893 total time=   4.1s
[CV 4/5; 21/24] START criterion=entropy, max_depth=5, n_esti
mators=1000.........
[CV 4/5; 21/24] END criterion=entropy, max_depth=5, n_estima
tors=1000;, score=0.893 total time=   4.5s
[CV 5/5; 21/24] START criterion=entropy, max_depth=5, n_esti
mators=1000.........
[CV 5/5; 21/24] END criterion=entropy, max_depth=5, n_estima
tors=1000;, score=0.893 total time=   4.3s
[CV 1/5; 22/24] START criterion=entropy, max_depth=7, n_esti
mators=100..........
[CV 1/5; 22/24] END criterion=entropy, max_depth=7, n_estima
tors=100;, score=0.893 total time=   0.7s
[CV 2/5; 22/24] START criterion=entropy, max_depth=7, n_esti
mators=100..........
[CV 2/5; 22/24] END criterion=entropy, max_depth=7, n_estima
tors=100;, score=0.889 total time=   0.8s
[CV 3/5; 22/24] START criterion=entropy, max_depth=7, n_esti
mators=100..........
[CV 3/5; 22/24] END criterion=entropy, max_depth=7, n_estima
tors=100;, score=0.893 total time=   0.6s
[CV 4/5; 22/24] START criterion=entropy, max_depth=7, n_esti
mators=100..........
[CV 4/5; 22/24] END criterion=entropy, max_depth=7, n_estima
tors=100;, score=0.894 total time=   0.6s
[CV 5/5; 22/24] START criterion=entropy, max_depth=7, n_esti
mators=100..........
[CV 5/5; 22/24] END criterion=entropy, max_depth=7, n_estima
tors=100;, score=0.893 total time=   0.6s
[CV 1/5; 23/24] START criterion=entropy, max_depth=7, n_esti
mators=500..........
[CV 1/5; 23/24] END criterion=entropy, max_depth=7, n_estima
tors=500;, score=0.893 total time=   2.6s
[CV 2/5; 23/24] START criterion=entropy, max_depth=7, n_esti
mators=500..........
[CV 2/5; 23/24] END criterion=entropy, max_depth=7, n_estima
tors=500;, score=0.890 total time=   2.6s
[CV 3/5; 23/24] START criterion=entropy, max_depth=7, n_esti
mators=500..........
[CV 3/5; 23/24] END criterion=entropy, max_depth=7, n_estima
tors=500;, score=0.893 total time=   3.0s
[CV 4/5; 23/24] START criterion=entropy, max_depth=7, n_esti
mators=500..........
```

```
[CV 4/5; 23/24] END criterion=entropy, max_depth=7, n_estima
tors=500;, score=0.896 total time=   2.5s
[CV 5/5; 23/24] START criterion=entropy, max_depth=7, n_esti
mators=500..........
[CV 5/5; 23/24] END criterion=entropy, max_depth=7, n_estima
tors=500;, score=0.893 total time=   2.6s
[CV 1/5; 24/24] START criterion=entropy, max_depth=7, n_esti
mators=1000.........
[CV 1/5; 24/24] END criterion=entropy, max_depth=7, n_estima
tors=1000;, score=0.893 total time=   4.7s
[CV 2/5; 24/24] START criterion=entropy, max_depth=7, n_esti
mators=1000.........
[CV 2/5; 24/24] END criterion=entropy, max_depth=7, n_estima
tors=1000;, score=0.890 total time=   5.2s
[CV 3/5; 24/24] START criterion=entropy, max_depth=7, n_esti
mators=1000.........
[CV 3/5; 24/24] END criterion=entropy, max_depth=7, n_estima
tors=1000;, score=0.893 total time=   5.6s
[CV 4/5; 24/24] START criterion=entropy, max_depth=7, n_esti
mators=1000.........
[CV 4/5; 24/24] END criterion=entropy, max_depth=7, n_estima
tors=1000;, score=0.895 total time=   5.0s
[CV 5/5; 24/24] START criterion=entropy, max_depth=7, n_esti
mators=1000.........
[CV 5/5; 24/24] END criterion=entropy, max_depth=7, n_estima
tors=1000;, score=0.893 total time=   5.0s
0.8928305345396799
{'bootstrap': True, 'ccp_alpha': 0.0, 'class_weight': None,
'criterion': 'entropy', 'max_depth': 7, 'max_features': 'sqr
t', 'max_leaf_nodes': None, 'max_samples': None, 'min_impuri
ty_decrease': 0.0, 'min_samples_leaf': 1, 'min_samples_spli
t': 2, 'min_weight_fraction_leaf': 0.0, 'n_estimators': 500,
'n_jobs': -1, 'oob_score': False, 'random_state': None, 'ver
bose': 0, 'warm_start': False}
```

In [6…
```
model3 = RandomForestClassifier(max_depth=7, min_samples_spli


Y_pred_test = model_fit_predict(model3, X_train, Y_train, X_t

#f1 score for training data
f1 = round(f1_score(Y_tests, Y_pred_test),2)

#accuracy score for training data
acc = round(accuracy_score(Y_tests, Y_pred_test),2)

#precision score for training data
pre = round(precision_score(Y_tests, Y_pred_test),2)

print(f"Accuracy, precision and f1-score for training data ar
```

```
Accuracy, precision and f1-score for training data are 0.9,
1.0 and 0.23 respectively
```

In [6…
```
predictions3 = model3.predict(X_tests)
tn, fp, fn, tp = metrics.confusion_matrix(Y_tests, predictior
Y_tests.value_counts()
```

Out[64]:
```
churn
0    3556
1     468
Name: count, dtype: int64
```

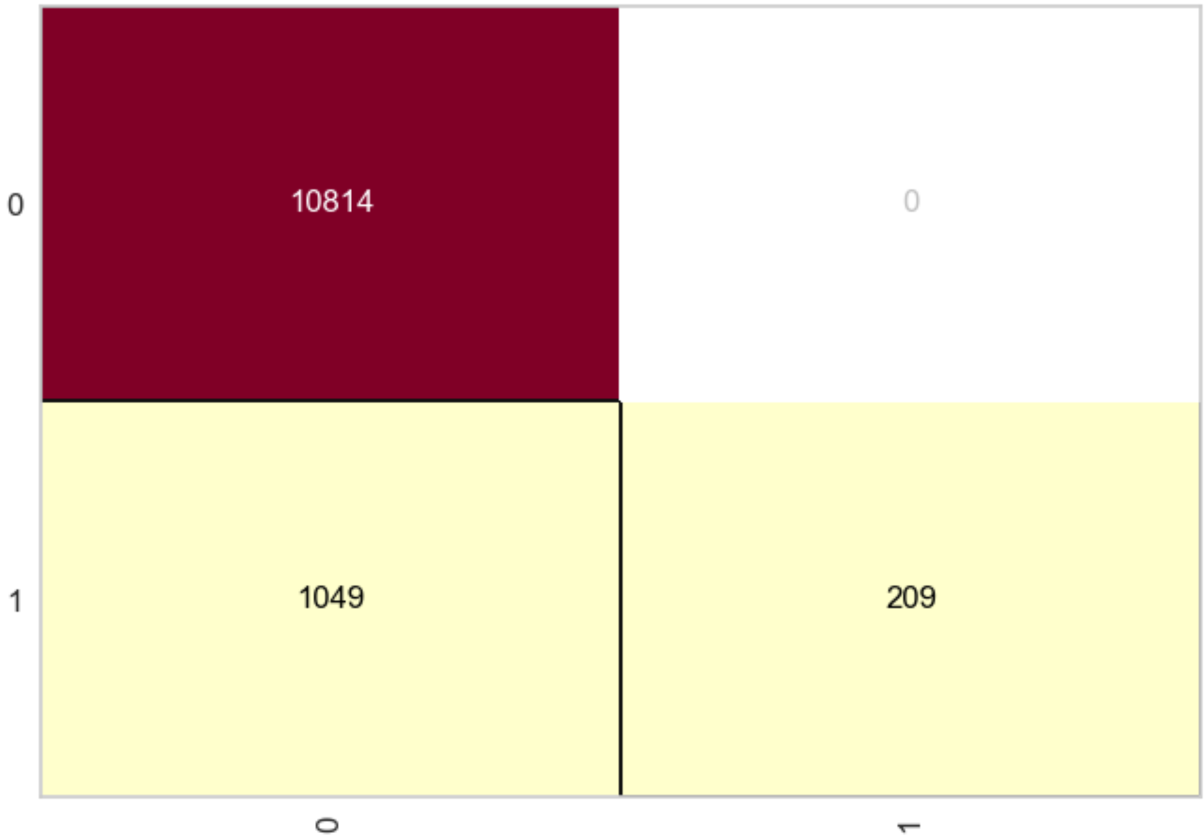
In [6…
```
print(f"True positives: {tp}")
print(f"False positives: {fp}")
print(f"True negatives: {tn}")
print(f"False negatives: {fn}\n")

print(f"Accuracy: {metrics.accuracy_score(Y_tests, predictior
print(f"Precision: {metrics.precision_score(Y_tests, predicti
print(f"Recall: {metrics.recall_score(Y_tests, predictions3)}
```

```
True positives: 60
False positives: 0
True negatives: 3556
False negatives: 408

Accuracy: 0.8986083499005965
Precision: 1.0
Recall: 0.1282051282051282
```

Model shows very low level of Precision, likely due to imbalanced churn in the data , leading to false predictions

In [6…
```
cm= ConfusionMatrix(model3, classes=[0,1])
cm.fit(X_train, Y_train)

cm.score(X_train, Y_train)
```

Out[66]: 0.913104705102717



Balancing Data using Random Sampling

In [6…
```
from imblearn.under_sampling import RandomUnderSampler
```

In [6…
```
count_class_0, count_class_1=model_df.churn.value_counts()
model_df_0=model_df[model_df['churn']==0]
model_df_1=model_df[model_df['churn']==1]
```

In [6…
```
count_class_0, count_class_1
```

Out[69]: (14370, 1726)

In [7…
```
model_df_0.shape
```

Out[70]: (14370, 37)

In [7…

```
model_df_0_under=model_df_0.sample(count_class_1)
USampledDf=pd.concat([model_df_0_under,model_df_1], axis=0)
USampledDf.head(5)
```

Out[71]:

	id	cons_12m	cons_gas_12m	co
5377	87ff1797c781c6f51bc199e83cc96b54	3.604334	0.000000	
14336	ce406580e6356c422f17a3c462788611	4.088703	0.000000	
2620	353c7a38a3fafd078eb86809031a7337	4.500936	0.000000	
10855	6e86731b7ff0fbff28702e21c01b8612	3.764027	0.000000	
9863	893a9debddaaf96213274575d3a68b72	6.550264	6.146552	

5 rows × 37 columns

In [7…

```
Y = USampledDf['churn']
X = USampledDf.drop(columns=['id', 'churn'])
X_train, X_tests, Y_train, Y_tests = train_test_split(X, Y, t
print ('Train set:', X_train.shape,  Y_train.shape)
print ('Test set:', X_tests.shape,  Y_tests.shape)
```

```
Train set: (2589, 35) (2589,)
Test set: (863, 35) (863,)
```

In [7…

```
modelS = RandomForestClassifier(max_depth=7, min_samples_spli


Y_pred_test = model_fit_predict(modelS, X_train, Y_train, X_t

#f1 score for training data
f1 = round(f1_score(Y_tests, Y_pred_test),2)

#accuracy score for training data
acc = round(accuracy_score(Y_tests, Y_pred_test),2)

#precision score for training data
pre = round(precision_score(Y_tests, Y_pred_test),2)

print(f"Accuracy, precision and f1-score for training data ar
```

```
Accuracy, precision and f1-score for training data are 0.92,
0.86 and 0.92 respectively
```

In […

In [7…

```
predictionss = modelS.predict(X_tests)
tn, fp, fn, tp = metrics.confusion_matrix(Y_tests, predictior
Y_tests.value_counts()

print(f"True positives: {tp}")
print(f"False positives: {fp}")
print(f"True negatives: {tn}")
print(f"False negatives: {fn}\n")

print(f"Accuracy: {metrics.accuracy_score(Y_tests, predictior
print(f"Precision: {metrics.precision_score(Y_tests, predicti
print(f"Recall: {metrics.recall_score(Y_tests, predictionss)}
```

```
True positives: 398
False positives: 66
True negatives: 393
False negatives: 6

Accuracy: 0.9165701042873696
Precision: 0.8577586206896551
Recall: 0.9851485148514851
```

In [7…

```
print(classification_report(Y_tests, Y_pred_test))
```

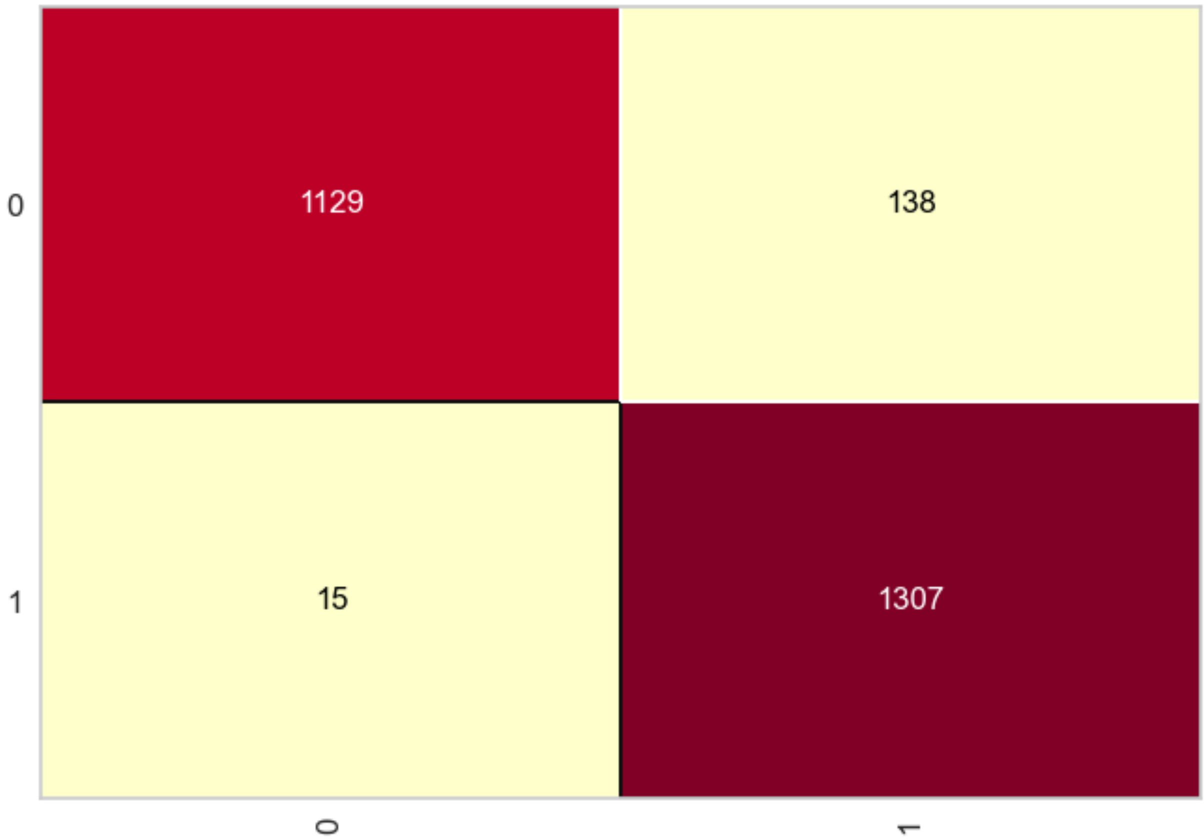
```
              precision    recall  f1-score   support

           0       0.98      0.86      0.92       459
           1       0.86      0.99      0.92       404

    accuracy                           0.92       863
   macro avg       0.92      0.92      0.92       863
weighted avg       0.93      0.92      0.92       863
```

In [7…

```
cm= ConfusionMatrix(modelS, classes=[0,1])
cm.fit(X_train, Y_train)

cm.score(X_train, Y_train)
```

Out[76]:

```
0.9409038238702202
```



In [7…

```
from sklearn import ensemble, model_selection

classifier = ensemble.RandomForestClassifier(n_jobs=-1)
param_grid= {
    'n_estimators':[100,500,1000],
    'max_depth':[1,3,5,7],
    'criterion':["gini","entropy"],

}

model=model_selection.GridSearchCV(
    estimator=classifier,
    param_grid=param_grid,
    scoring="accuracy",
    verbose=10,
    n_jobs=1,
    cv=5,
)

model.fit(X,Y)
print(model.best_score_)
print(model.best_estimator_.get_params())
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fit
s
[CV 1/5; 1/24] START criterion=gini, max_depth=1, n_estimato
```

```
rs=100.............
[CV 1/5; 1/24] END criterion=gini, max_depth=1, n_estimators
=100;, score=0.648 total time=   0.4s
[CV 2/5; 1/24] START criterion=gini, max_depth=1, n_estimato
rs=100.............
[CV 2/5; 1/24] END criterion=gini, max_depth=1, n_estimators
=100;, score=0.576 total time=   0.2s
[CV 3/5; 1/24] START criterion=gini, max_depth=1, n_estimato
rs=100.............
[CV 3/5; 1/24] END criterion=gini, max_depth=1, n_estimators
=100;, score=0.674 total time=   0.2s
[CV 4/5; 1/24] START criterion=gini, max_depth=1, n_estimato
rs=100.............
[CV 4/5; 1/24] END criterion=gini, max_depth=1, n_estimators
=100;, score=0.691 total time=   0.2s
[CV 5/5; 1/24] START criterion=gini, max_depth=1, n_estimato
rs=100.............
[CV 5/5; 1/24] END criterion=gini, max_depth=1, n_estimators
=100;, score=0.580 total time=   0.2s
[CV 1/5; 2/24] START criterion=gini, max_depth=1, n_estimato
rs=500.............
[CV 1/5; 2/24] END criterion=gini, max_depth=1, n_estimators
=500;, score=0.660 total time=   0.7s
[CV 2/5; 2/24] START criterion=gini, max_depth=1, n_estimato
rs=500.............
[CV 2/5; 2/24] END criterion=gini, max_depth=1, n_estimators
=500;, score=0.588 total time=   0.7s
[CV 3/5; 2/24] START criterion=gini, max_depth=1, n_estimato
rs=500.............
[CV 3/5; 2/24] END criterion=gini, max_depth=1, n_estimators
=500;, score=0.607 total time=   0.8s
[CV 4/5; 2/24] START criterion=gini, max_depth=1, n_estimato
rs=500.............
[CV 4/5; 2/24] END criterion=gini, max_depth=1, n_estimators
=500;, score=0.672 total time=   0.8s
[CV 5/5; 2/24] START criterion=gini, max_depth=1, n_estimato
rs=500.............
[CV 5/5; 2/24] END criterion=gini, max_depth=1, n_estimators
=500;, score=0.586 total time=   0.7s
[CV 1/5; 3/24] START criterion=gini, max_depth=1, n_estimato
rs=1000............
[CV 1/5; 3/24] END criterion=gini, max_depth=1, n_estimators
=1000;, score=0.674 total time=   1.8s
[CV 2/5; 3/24] START criterion=gini, max_depth=1, n_estimato
rs=1000............
[CV 2/5; 3/24] END criterion=gini, max_depth=1, n_estimators
=1000;, score=0.595 total time=   1.8s
[CV 3/5; 3/24] START criterion=gini, max_depth=1, n_estimato
```

```
rs=1000.............
[CV 3/5; 3/24] END criterion=gini, max_depth=1, n_estimators
=1000;, score=0.616 total time=   1.4s
[CV 4/5; 3/24] START criterion=gini, max_depth=1, n_estimato
rs=1000.............
[CV 4/5; 3/24] END criterion=gini, max_depth=1, n_estimators
=1000;, score=0.684 total time=   1.6s
[CV 5/5; 3/24] START criterion=gini, max_depth=1, n_estimato
rs=1000.............
[CV 5/5; 3/24] END criterion=gini, max_depth=1, n_estimators
=1000;, score=0.586 total time=   1.5s
[CV 1/5; 4/24] START criterion=gini, max_depth=3, n_estimato
rs=100..............
[CV 1/5; 4/24] END criterion=gini, max_depth=3, n_estimators
=100;, score=0.635 total time=   0.2s
[CV 2/5; 4/24] START criterion=gini, max_depth=3, n_estimato
rs=100..............
[CV 2/5; 4/24] END criterion=gini, max_depth=3, n_estimators
=100;, score=0.621 total time=   0.2s
[CV 3/5; 4/24] START criterion=gini, max_depth=3, n_estimato
rs=100..............
[CV 3/5; 4/24] END criterion=gini, max_depth=3, n_estimators
=100;, score=0.635 total time=   0.2s
[CV 4/5; 4/24] START criterion=gini, max_depth=3, n_estimato
rs=100..............
[CV 4/5; 4/24] END criterion=gini, max_depth=3, n_estimators
=100;, score=0.652 total time=   0.2s
[CV 5/5; 4/24] START criterion=gini, max_depth=3, n_estimato
rs=100..............
[CV 5/5; 4/24] END criterion=gini, max_depth=3, n_estimators
=100;, score=0.626 total time=   0.4s
[CV 1/5; 5/24] START criterion=gini, max_depth=3, n_estimato
rs=500..............
[CV 1/5; 5/24] END criterion=gini, max_depth=3, n_estimators
=500;, score=0.669 total time=   1.0s
[CV 2/5; 5/24] START criterion=gini, max_depth=3, n_estimato
rs=500..............
[CV 2/5; 5/24] END criterion=gini, max_depth=3, n_estimators
=500;, score=0.589 total time=   0.8s
[CV 3/5; 5/24] START criterion=gini, max_depth=3, n_estimato
rs=500..............
[CV 3/5; 5/24] END criterion=gini, max_depth=3, n_estimators
=500;, score=0.670 total time=   0.9s
[CV 4/5; 5/24] START criterion=gini, max_depth=3, n_estimato
rs=500..............
[CV 4/5; 5/24] END criterion=gini, max_depth=3, n_estimators
=500;, score=0.645 total time=   0.9s
[CV 5/5; 5/24] START criterion=gini, max_depth=3, n_estimato
```

```
rs=500.............
[CV 5/5; 5/24] END criterion=gini, max_depth=3, n_estimators
=500;, score=0.633 total time=   0.9s
[CV 1/5; 6/24] START criterion=gini, max_depth=3, n_estimato
rs=1000............
[CV 1/5; 6/24] END criterion=gini, max_depth=3, n_estimators
=1000;, score=0.666 total time=   1.6s
[CV 2/5; 6/24] START criterion=gini, max_depth=3, n_estimato
rs=1000............
[CV 2/5; 6/24] END criterion=gini, max_depth=3, n_estimators
=1000;, score=0.616 total time=   1.7s
[CV 3/5; 6/24] START criterion=gini, max_depth=3, n_estimato
rs=1000............
[CV 3/5; 6/24] END criterion=gini, max_depth=3, n_estimators
=1000;, score=0.665 total time=   1.6s
[CV 4/5; 6/24] START criterion=gini, max_depth=3, n_estimato
rs=1000............
[CV 4/5; 6/24] END criterion=gini, max_depth=3, n_estimators
=1000;, score=0.643 total time=   1.8s
[CV 5/5; 6/24] START criterion=gini, max_depth=3, n_estimato
rs=1000............
[CV 5/5; 6/24] END criterion=gini, max_depth=3, n_estimators
=1000;, score=0.628 total time=   2.0s
[CV 1/5; 7/24] START criterion=gini, max_depth=5, n_estimato
rs=100.............
[CV 1/5; 7/24] END criterion=gini, max_depth=5, n_estimators
=100;, score=0.661 total time=   0.2s
[CV 2/5; 7/24] START criterion=gini, max_depth=5, n_estimato
rs=100.............
[CV 2/5; 7/24] END criterion=gini, max_depth=5, n_estimators
=100;, score=0.663 total time=   0.3s
[CV 3/5; 7/24] START criterion=gini, max_depth=5, n_estimato
rs=100.............
[CV 3/5; 7/24] END criterion=gini, max_depth=5, n_estimators
=100;, score=0.657 total time=   0.3s
[CV 4/5; 7/24] START criterion=gini, max_depth=5, n_estimato
rs=100.............
[CV 4/5; 7/24] END criterion=gini, max_depth=5, n_estimators
=100;, score=0.671 total time=   0.3s
[CV 5/5; 7/24] START criterion=gini, max_depth=5, n_estimato
rs=100.............
[CV 5/5; 7/24] END criterion=gini, max_depth=5, n_estimators
=100;, score=0.636 total time=   0.3s
[CV 1/5; 8/24] START criterion=gini, max_depth=5, n_estimato
rs=500.............
[CV 1/5; 8/24] END criterion=gini, max_depth=5, n_estimators
=500;, score=0.682 total time=   1.0s
[CV 2/5; 8/24] START criterion=gini, max_depth=5, n_estimato
```

```
rs=500.............
[CV 2/5; 8/24] END criterion=gini, max_depth=5, n_estimators
=500;, score=0.631 total time=   1.0s
[CV 3/5; 8/24] START criterion=gini, max_depth=5, n_estimato
rs=500.............
[CV 3/5; 8/24] END criterion=gini, max_depth=5, n_estimators
=500;, score=0.668 total time=   1.1s
[CV 4/5; 8/24] START criterion=gini, max_depth=5, n_estimato
rs=500.............
[CV 4/5; 8/24] END criterion=gini, max_depth=5, n_estimators
=500;, score=0.683 total time=   1.0s
[CV 5/5; 8/24] START criterion=gini, max_depth=5, n_estimato
rs=500.............
[CV 5/5; 8/24] END criterion=gini, max_depth=5, n_estimators
=500;, score=0.633 total time=   1.0s
[CV 1/5; 9/24] START criterion=gini, max_depth=5, n_estimato
rs=1000............
[CV 1/5; 9/24] END criterion=gini, max_depth=5, n_estimators
=1000;, score=0.644 total time=   1.9s
[CV 2/5; 9/24] START criterion=gini, max_depth=5, n_estimato
rs=1000............
[CV 2/5; 9/24] END criterion=gini, max_depth=5, n_estimators
=1000;, score=0.638 total time=   1.8s
[CV 3/5; 9/24] START criterion=gini, max_depth=5, n_estimato
rs=1000............
[CV 3/5; 9/24] END criterion=gini, max_depth=5, n_estimators
=1000;, score=0.655 total time=   1.9s
[CV 4/5; 9/24] START criterion=gini, max_depth=5, n_estimato
rs=1000............
[CV 4/5; 9/24] END criterion=gini, max_depth=5, n_estimators
=1000;, score=0.671 total time=   2.6s
[CV 5/5; 9/24] START criterion=gini, max_depth=5, n_estimato
rs=1000............
[CV 5/5; 9/24] END criterion=gini, max_depth=5, n_estimators
=1000;, score=0.645 total time=   2.6s
[CV 1/5; 10/24] START criterion=gini, max_depth=7, n_estimat
ors=100............
[CV 1/5; 10/24] END criterion=gini, max_depth=7, n_estimator
s=100;, score=0.643 total time=   0.3s
[CV 2/5; 10/24] START criterion=gini, max_depth=7, n_estimat
ors=100............
[CV 2/5; 10/24] END criterion=gini, max_depth=7, n_estimator
s=100;, score=0.640 total time=   0.3s
[CV 3/5; 10/24] START criterion=gini, max_depth=7, n_estimat
ors=100............
[CV 3/5; 10/24] END criterion=gini, max_depth=7, n_estimator
s=100;, score=0.659 total time=   0.3s
[CV 4/5; 10/24] START criterion=gini, max_depth=7, n_estimat
```

```
ors=100............
[CV 4/5; 10/24] END criterion=gini, max_depth=7, n_estimator
s=100;, score=0.604 total time=   0.3s
[CV 5/5; 10/24] START criterion=gini, max_depth=7, n_estimat
ors=100............
[CV 5/5; 10/24] END criterion=gini, max_depth=7, n_estimator
s=100;, score=0.662 total time=   0.3s
[CV 1/5; 11/24] START criterion=gini, max_depth=7, n_estimat
ors=500............
[CV 1/5; 11/24] END criterion=gini, max_depth=7, n_estimator
s=500;, score=0.638 total time=   1.1s
[CV 2/5; 11/24] START criterion=gini, max_depth=7, n_estimat
ors=500............
[CV 2/5; 11/24] END criterion=gini, max_depth=7, n_estimator
s=500;, score=0.645 total time=   1.1s
[CV 3/5; 11/24] START criterion=gini, max_depth=7, n_estimat
ors=500............
[CV 3/5; 11/24] END criterion=gini, max_depth=7, n_estimator
s=500;, score=0.651 total time=   1.1s
[CV 4/5; 11/24] START criterion=gini, max_depth=7, n_estimat
ors=500............
[CV 4/5; 11/24] END criterion=gini, max_depth=7, n_estimator
s=500;, score=0.664 total time=   1.1s
[CV 5/5; 11/24] START criterion=gini, max_depth=7, n_estimat
ors=500............
[CV 5/5; 11/24] END criterion=gini, max_depth=7, n_estimator
s=500;, score=0.655 total time=   1.1s
[CV 1/5; 12/24] START criterion=gini, max_depth=7, n_estimat
ors=1000...........
[CV 1/5; 12/24] END criterion=gini, max_depth=7, n_estimator
s=1000;, score=0.637 total time=   2.1s
[CV 2/5; 12/24] START criterion=gini, max_depth=7, n_estimat
ors=1000...........
[CV 2/5; 12/24] END criterion=gini, max_depth=7, n_estimator
s=1000;, score=0.648 total time=   2.4s
[CV 3/5; 12/24] START criterion=gini, max_depth=7, n_estimat
ors=1000...........
[CV 3/5; 12/24] END criterion=gini, max_depth=7, n_estimator
s=1000;, score=0.641 total time=   3.6s
[CV 4/5; 12/24] START criterion=gini, max_depth=7, n_estimat
ors=1000...........
[CV 4/5; 12/24] END criterion=gini, max_depth=7, n_estimator
s=1000;, score=0.648 total time=   2.7s
[CV 5/5; 12/24] START criterion=gini, max_depth=7, n_estimat
ors=1000...........
[CV 5/5; 12/24] END criterion=gini, max_depth=7, n_estimator
s=1000;, score=0.625 total time=   3.8s
[CV 1/5; 13/24] START criterion=entropy, max_depth=1, n_esti
```

```
mators=100..........
[CV 1/5; 13/24] END criterion=entropy, max_depth=1, n_estima
tors=100;, score=0.637 total time=   0.4s
[CV 2/5; 13/24] START criterion=entropy, max_depth=1, n_esti
mators=100..........
[CV 2/5; 13/24] END criterion=entropy, max_depth=1, n_estima
tors=100;, score=0.573 total time=   0.7s
[CV 3/5; 13/24] START criterion=entropy, max_depth=1, n_esti
mators=100..........
[CV 3/5; 13/24] END criterion=entropy, max_depth=1, n_estima
tors=100;, score=0.612 total time=   0.5s
[CV 4/5; 13/24] START criterion=entropy, max_depth=1, n_esti
mators=100..........
[CV 4/5; 13/24] END criterion=entropy, max_depth=1, n_estima
tors=100;, score=0.681 total time=   0.8s
[CV 5/5; 13/24] START criterion=entropy, max_depth=1, n_esti
mators=100..........
[CV 5/5; 13/24] END criterion=entropy, max_depth=1, n_estima
tors=100;, score=0.572 total time=   0.5s
[CV 1/5; 14/24] START criterion=entropy, max_depth=1, n_esti
mators=500..........
[CV 1/5; 14/24] END criterion=entropy, max_depth=1, n_estima
tors=500;, score=0.674 total time=   1.5s
[CV 2/5; 14/24] START criterion=entropy, max_depth=1, n_esti
mators=500..........
[CV 2/5; 14/24] END criterion=entropy, max_depth=1, n_estima
tors=500;, score=0.557 total time=   2.1s
[CV 3/5; 14/24] START criterion=entropy, max_depth=1, n_esti
mators=500..........
[CV 3/5; 14/24] END criterion=entropy, max_depth=1, n_estima
tors=500;, score=0.606 total time=   2.0s
[CV 4/5; 14/24] START criterion=entropy, max_depth=1, n_esti
mators=500..........
[CV 4/5; 14/24] END criterion=entropy, max_depth=1, n_estima
tors=500;, score=0.680 total time=   1.7s
[CV 5/5; 14/24] START criterion=entropy, max_depth=1, n_esti
mators=500..........
[CV 5/5; 14/24] END criterion=entropy, max_depth=1, n_estima
tors=500;, score=0.583 total time=   0.9s
[CV 1/5; 15/24] START criterion=entropy, max_depth=1, n_esti
mators=1000.........
[CV 1/5; 15/24] END criterion=entropy, max_depth=1, n_estima
tors=1000;, score=0.673 total time=   1.4s
[CV 2/5; 15/24] START criterion=entropy, max_depth=1, n_esti
mators=1000.........
[CV 2/5; 15/24] END criterion=entropy, max_depth=1, n_estima
tors=1000;, score=0.557 total time=   1.3s
[CV 3/5; 15/24] START criterion=entropy, max_depth=1, n_esti
```

```
mators=1000.........
[CV 3/5; 15/24] END criterion=entropy, max_depth=1, n_estima
tors=1000;, score=0.628 total time=   1.4s
[CV 4/5; 15/24] START criterion=entropy, max_depth=1, n_esti
mators=1000.........
[CV 4/5; 15/24] END criterion=entropy, max_depth=1, n_estima
tors=1000;, score=0.687 total time=   1.4s
[CV 5/5; 15/24] START criterion=entropy, max_depth=1, n_esti
mators=1000.........
[CV 5/5; 15/24] END criterion=entropy, max_depth=1, n_estima
tors=1000;, score=0.586 total time=   1.3s
[CV 1/5; 16/24] START criterion=entropy, max_depth=3, n_esti
mators=100..........
[CV 1/5; 16/24] END criterion=entropy, max_depth=3, n_estima
tors=100;, score=0.614 total time=   0.2s
[CV 2/5; 16/24] START criterion=entropy, max_depth=3, n_esti
mators=100..........
[CV 2/5; 16/24] END criterion=entropy, max_depth=3, n_estima
tors=100;, score=0.606 total time=   0.3s
[CV 3/5; 16/24] START criterion=entropy, max_depth=3, n_esti
mators=100..........
[CV 3/5; 16/24] END criterion=entropy, max_depth=3, n_estima
tors=100;, score=0.667 total time=   0.2s
[CV 4/5; 16/24] START criterion=entropy, max_depth=3, n_esti
mators=100..........
[CV 4/5; 16/24] END criterion=entropy, max_depth=3, n_estima
tors=100;, score=0.652 total time=   0.2s
[CV 5/5; 16/24] START criterion=entropy, max_depth=3, n_esti
mators=100..........
[CV 5/5; 16/24] END criterion=entropy, max_depth=3, n_estima
tors=100;, score=0.607 total time=   0.3s
[CV 1/5; 17/24] START criterion=entropy, max_depth=3, n_esti
mators=500..........
[CV 1/5; 17/24] END criterion=entropy, max_depth=3, n_estima
tors=500;, score=0.664 total time=   1.3s
[CV 2/5; 17/24] START criterion=entropy, max_depth=3, n_esti
mators=500..........
[CV 2/5; 17/24] END criterion=entropy, max_depth=3, n_estima
tors=500;, score=0.616 total time=   1.3s
[CV 3/5; 17/24] START criterion=entropy, max_depth=3, n_esti
mators=500..........
[CV 3/5; 17/24] END criterion=entropy, max_depth=3, n_estima
tors=500;, score=0.671 total time=   0.9s
[CV 4/5; 17/24] START criterion=entropy, max_depth=3, n_esti
mators=500..........
[CV 4/5; 17/24] END criterion=entropy, max_depth=3, n_estima
tors=500;, score=0.643 total time=   0.9s
[CV 5/5; 17/24] START criterion=entropy, max_depth=3, n_esti
```

```
mators=500..........
[CV 5/5; 17/24] END criterion=entropy, max_depth=3, n_estima
tors=500;, score=0.617 total time=   0.9s
[CV 1/5; 18/24] START criterion=entropy, max_depth=3, n_esti
mators=1000.........
[CV 1/5; 18/24] END criterion=entropy, max_depth=3, n_estima
tors=1000;, score=0.648 total time=   1.7s
[CV 2/5; 18/24] START criterion=entropy, max_depth=3, n_esti
mators=1000.........
[CV 2/5; 18/24] END criterion=entropy, max_depth=3, n_estima
tors=1000;, score=0.616 total time=   1.7s
[CV 3/5; 18/24] START criterion=entropy, max_depth=3, n_esti
mators=1000.........
[CV 3/5; 18/24] END criterion=entropy, max_depth=3, n_estima
tors=1000;, score=0.654 total time=   1.7s
[CV 4/5; 18/24] START criterion=entropy, max_depth=3, n_esti
mators=1000.........
[CV 4/5; 18/24] END criterion=entropy, max_depth=3, n_estima
tors=1000;, score=0.645 total time=   1.7s
[CV 5/5; 18/24] START criterion=entropy, max_depth=3, n_esti
mators=1000.........
[CV 5/5; 18/24] END criterion=entropy, max_depth=3, n_estima
tors=1000;, score=0.633 total time=   1.6s
[CV 1/5; 19/24] START criterion=entropy, max_depth=5, n_esti
mators=100..........
[CV 1/5; 19/24] END criterion=entropy, max_depth=5, n_estima
tors=100;, score=0.611 total time=   0.3s
[CV 2/5; 19/24] START criterion=entropy, max_depth=5, n_esti
mators=100..........
[CV 2/5; 19/24] END criterion=entropy, max_depth=5, n_estima
tors=100;, score=0.614 total time=   0.3s
[CV 3/5; 19/24] START criterion=entropy, max_depth=5, n_esti
mators=100..........
[CV 3/5; 19/24] END criterion=entropy, max_depth=5, n_estima
tors=100;, score=0.670 total time=   0.3s
[CV 4/5; 19/24] START criterion=entropy, max_depth=5, n_esti
mators=100..........
[CV 4/5; 19/24] END criterion=entropy, max_depth=5, n_estima
tors=100;, score=0.680 total time=   0.3s
[CV 5/5; 19/24] START criterion=entropy, max_depth=5, n_esti
mators=100..........
[CV 5/5; 19/24] END criterion=entropy, max_depth=5, n_estima
tors=100;, score=0.613 total time=   0.3s
[CV 1/5; 20/24] START criterion=entropy, max_depth=5, n_esti
mators=500..........
[CV 1/5; 20/24] END criterion=entropy, max_depth=5, n_estima
tors=500;, score=0.644 total time=   1.3s
[CV 2/5; 20/24] START criterion=entropy, max_depth=5, n_esti
```

```
mators=500..........
[CV 2/5; 20/24] END criterion=entropy, max_depth=5, n_estima
tors=500;, score=0.627 total time=   1.1s
[CV 3/5; 20/24] START criterion=entropy, max_depth=5, n_esti
mators=500..........
[CV 3/5; 20/24] END criterion=entropy, max_depth=5, n_estima
tors=500;, score=0.680 total time=   1.1s
[CV 4/5; 20/24] START criterion=entropy, max_depth=5, n_esti
mators=500..........
[CV 4/5; 20/24] END criterion=entropy, max_depth=5, n_estima
tors=500;, score=0.681 total time=   1.5s
[CV 5/5; 20/24] START criterion=entropy, max_depth=5, n_esti
mators=500..........
[CV 5/5; 20/24] END criterion=entropy, max_depth=5, n_estima
tors=500;, score=0.632 total time=   1.2s
[CV 1/5; 21/24] START criterion=entropy, max_depth=5, n_esti
mators=1000.........
[CV 1/5; 21/24] END criterion=entropy, max_depth=5, n_estima
tors=1000;, score=0.645 total time=   2.1s
[CV 2/5; 21/24] START criterion=entropy, max_depth=5, n_esti
mators=1000.........
[CV 2/5; 21/24] END criterion=entropy, max_depth=5, n_estima
tors=1000;, score=0.641 total time=   2.4s
[CV 3/5; 21/24] START criterion=entropy, max_depth=5, n_esti
mators=1000.........
[CV 3/5; 21/24] END criterion=entropy, max_depth=5, n_estima
tors=1000;, score=0.678 total time=   2.1s
[CV 4/5; 21/24] START criterion=entropy, max_depth=5, n_esti
mators=1000.........
[CV 4/5; 21/24] END criterion=entropy, max_depth=5, n_estima
tors=1000;, score=0.686 total time=   2.0s
[CV 5/5; 21/24] START criterion=entropy, max_depth=5, n_esti
mators=1000.........
[CV 5/5; 21/24] END criterion=entropy, max_depth=5, n_estima
tors=1000;, score=0.639 total time=   2.1s
[CV 1/5; 22/24] START criterion=entropy, max_depth=7, n_esti
mators=100..........
[CV 1/5; 22/24] END criterion=entropy, max_depth=7, n_estima
tors=100;, score=0.627 total time=   0.3s
[CV 2/5; 22/24] START criterion=entropy, max_depth=7, n_esti
mators=100..........
[CV 2/5; 22/24] END criterion=entropy, max_depth=7, n_estima
tors=100;, score=0.643 total time=   0.3s
[CV 3/5; 22/24] START criterion=entropy, max_depth=7, n_esti
mators=100..........
[CV 3/5; 22/24] END criterion=entropy, max_depth=7, n_estima
tors=100;, score=0.633 total time=   0.3s
[CV 4/5; 22/24] START criterion=entropy, max_depth=7, n_esti
```

```
mators=100..........
[CV 4/5; 22/24] END criterion=entropy, max_depth=7, n_estima
tors=100;, score=0.668 total time=   0.3s
[CV 5/5; 22/24] START criterion=entropy, max_depth=7, n_esti
mators=100..........
[CV 5/5; 22/24] END criterion=entropy, max_depth=7, n_estima
tors=100;, score=0.657 total time=   0.3s
[CV 1/5; 23/24] START criterion=entropy, max_depth=7, n_esti
mators=500..........
[CV 1/5; 23/24] END criterion=entropy, max_depth=7, n_estima
tors=500;, score=0.632 total time=   1.2s
[CV 2/5; 23/24] START criterion=entropy, max_depth=7, n_esti
mators=500..........
[CV 2/5; 23/24] END criterion=entropy, max_depth=7, n_estima
tors=500;, score=0.635 total time=   1.3s
[CV 3/5; 23/24] START criterion=entropy, max_depth=7, n_esti
mators=500..........
[CV 3/5; 23/24] END criterion=entropy, max_depth=7, n_estima
tors=500;, score=0.658 total time=   1.2s
[CV 4/5; 23/24] START criterion=entropy, max_depth=7, n_esti
mators=500..........
[CV 4/5; 23/24] END criterion=entropy, max_depth=7, n_estima
tors=500;, score=0.651 total time=   1.2s
[CV 5/5; 23/24] START criterion=entropy, max_depth=7, n_esti
mators=500..........
[CV 5/5; 23/24] END criterion=entropy, max_depth=7, n_estima
tors=500;, score=0.638 total time=   1.2s
[CV 1/5; 24/24] START criterion=entropy, max_depth=7, n_esti
mators=1000.........
[CV 1/5; 24/24] END criterion=entropy, max_depth=7, n_estima
tors=1000;, score=0.630 total time=   2.9s
[CV 2/5; 24/24] START criterion=entropy, max_depth=7, n_esti
mators=1000.........
[CV 2/5; 24/24] END criterion=entropy, max_depth=7, n_estima
tors=1000;, score=0.641 total time=   3.2s
[CV 3/5; 24/24] START criterion=entropy, max_depth=7, n_esti
mators=1000.........
[CV 3/5; 24/24] END criterion=entropy, max_depth=7, n_estima
tors=1000;, score=0.664 total time=   2.6s
[CV 4/5; 24/24] START criterion=entropy, max_depth=7, n_esti
mators=1000.........
[CV 4/5; 24/24] END criterion=entropy, max_depth=7, n_estima
tors=1000;, score=0.668 total time=   2.4s
[CV 5/5; 24/24] START criterion=entropy, max_depth=7, n_esti
mators=1000.........
[CV 5/5; 24/24] END criterion=entropy, max_depth=7, n_estima
tors=1000;, score=0.641 total time=   2.4s
0.6593296839279347
```

```
{'bootstrap': True, 'ccp_alpha': 0.0, 'class_weight': None,
'criterion': 'gini', 'max_depth': 5, 'max_features': 'sqrt',
'max_leaf_nodes': None, 'max_samples': None, 'min_impurity_d
ecrease': 0.0, 'min_samples_leaf': 1, 'min_samples_split':
2, 'min_weight_fraction_leaf': 0.0, 'n_estimators': 500, 'n_
jobs': -1, 'oob_score': False, 'random_state': None, 'verbos
e': 0, 'warm_start': False}
```

In [7…

```
modelS = RandomForestClassifier(max_depth=7, min_samples_spli


Y_pred_test = model_fit_predict(modelS, X_train, Y_train, X_t

#f1 score for training data
f1 = round(f1_score(Y_tests, Y_pred_test),2)

#accuracy score for training data
acc = round(accuracy_score(Y_tests, Y_pred_test),2)

#precision score for training data
pre = round(precision_score(Y_tests, Y_pred_test),2)

print(f"Accuracy, precision and f1-score for training data ar
```

```
Accuracy, precision and f1-score for training data are 0.92,
0.86 and 0.92 respectively
```

In [7…

```
predictionss = modelS.predict(X_tests)
tn, fp, fn, tp = metrics.confusion_matrix(Y_tests, predictior
Y_tests.value_counts()

print(f"True positives: {tp}")
print(f"False positives: {fp}")
print(f"True negatives: {tn}")
print(f"False negatives: {fn}\n")

print(f"Accuracy: {metrics.accuracy_score(Y_tests, predictior
print(f"Precision: {metrics.precision_score(Y_tests, predicti
print(f"Recall: {metrics.recall_score(Y_tests, predictionss)}
```

```
True positives: 403
False positives: 66
True negatives: 393
False negatives: 1

Accuracy: 0.9223638470451911
Precision: 0.8592750533049041
Recall: 0.9975247524752475
```

In [8…

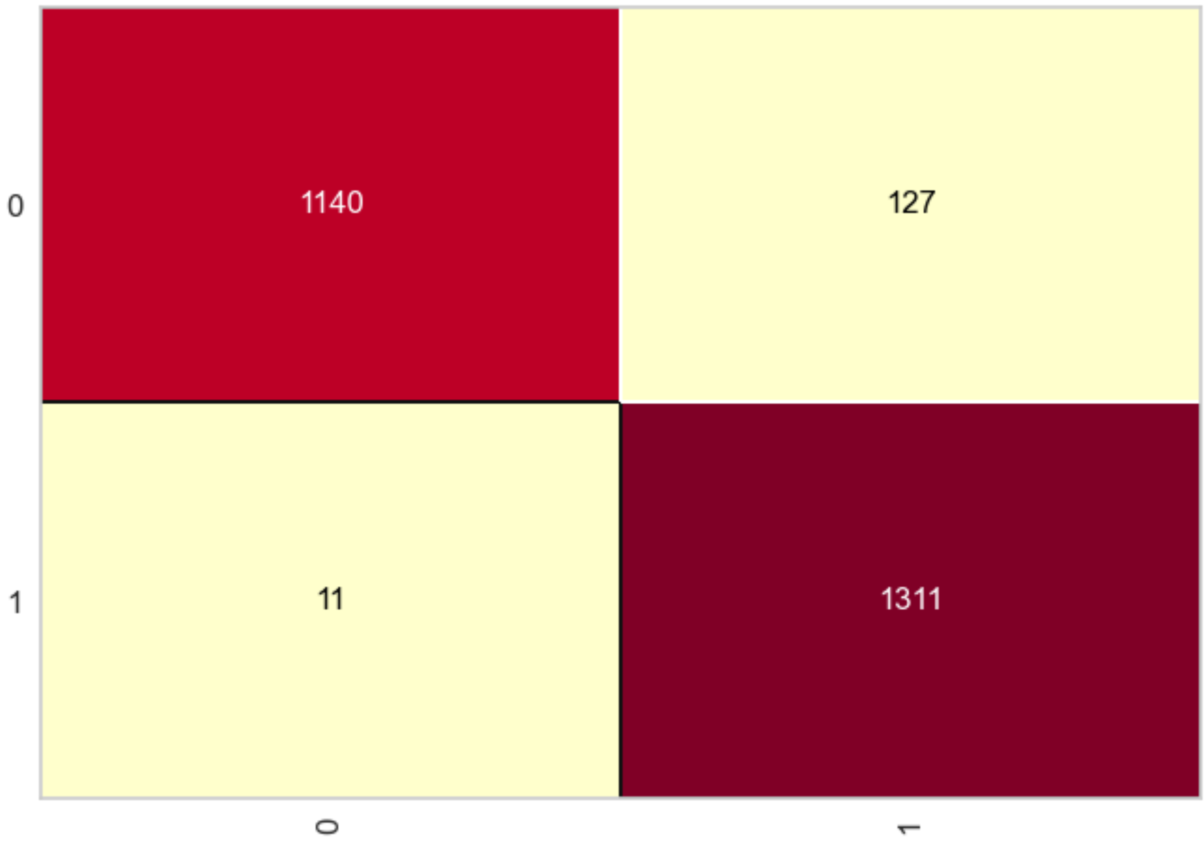
```
print(classification_report(Y_tests, Y_pred_test))
```

```
              precision    recall  f1-score   support

           0       1.00      0.86      0.92       459
           1       0.86      1.00      0.92       404

    accuracy                           0.92       863
   macro avg       0.93      0.93      0.92       863
weighted avg       0.93      0.92      0.92       863
```

In [8…

```
cm= ConfusionMatrix(modelS, classes=[0,1])
cm.fit(X_train, Y_train)

cm.score(X_train, Y_train)
```

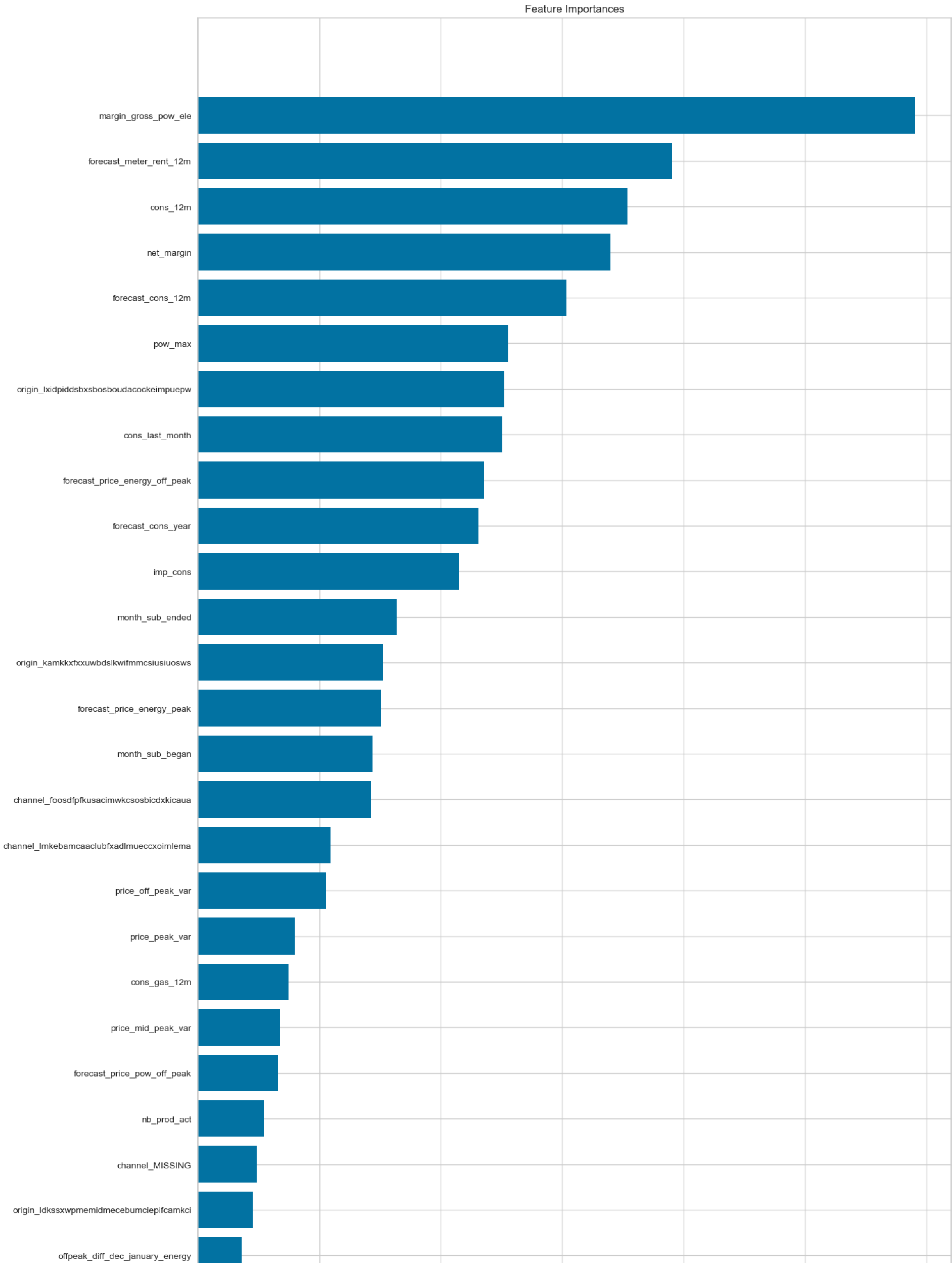
Out[81]: 0.9466975666280417

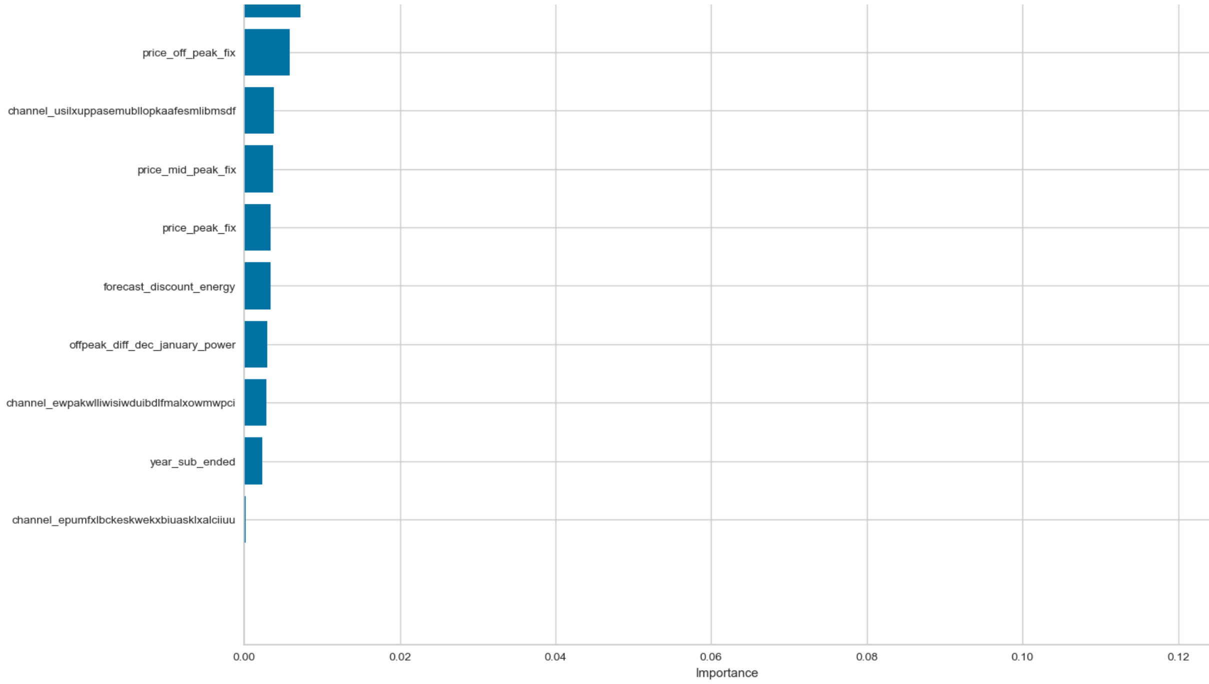


In […

In [8…

```
feature_importances = pd.DataFrame({'features': X_train.colum
plt.figure(figsize=(15, 35))
plt.title('Feature Importances')
plt.barh(range(len(feature_importances)), feature_importances
plt.yticks(range(len(feature_importances)), feature_importanc
plt.xlabel('Importance')
plt.show()
```





In [8…

```
print("Training Data Columns:", X_train.columns)
print("Testing Data Columns:", X_tests.columns)
```

```
Training Data Columns: Index(['cons_12m', 'cons_gas_12m', 'c
ons_last_month', 'forecast_cons_12m',
       'forecast_cons_year', 'forecast_discount_energy',
       'forecast_meter_rent_12m', 'forecast_price_energy_off
_peak',
       'forecast_price_energy_peak', 'forecast_price_pow_off
_peak', 'imp_cons',
       'margin_gross_pow_ele', 'nb_prod_act', 'net_margin',
'pow_max',
       'price_off_peak_var', 'price_peak_var', 'price_mid_pe
ak_var',
       'price_off_peak_fix', 'price_peak_fix', 'price_mid_pe
ak_fix',
       'offpeak_diff_dec_january_energy', 'offpeak_diff_dec_
january_power',
       'month_sub_ended', 'year_sub_ended', 'month_sub_bega
n',
       'channel_MISSING', 'channel_epumfxlbckeskwekxbiuasklx
alciiuu',
       'channel_ewpakwlliwisiwduibdlfmalxowmwpci',
       'channel_foosdfpfkusacimwkcsosbicdxkicaua',
       'channel_lmkebamcaaclubfxadlmueccxoimlema',
       'channel_usilxuppasemubllopkaafesmlibmsdf',
       'origin_kamkkxfxxuwbdslkwifmmcsiusiuosws',
       'origin_ldkssxwpmemidmecebumciepifcamkci',
       'origin_lxidpiddsbxsbosboudacockeimpuepw'],
      dtype='object')
```

```
Testing Data Columns: Index(['cons_12m', 'cons_gas_12m', 'co
ns_last_month', 'forecast_cons_12m',
       'forecast_cons_year', 'forecast_discount_energy',
       'forecast_meter_rent_12m', 'forecast_price_energy_off
_peak',
       'forecast_price_energy_peak', 'forecast_price_pow_off
_peak', 'imp_cons',
       'margin_gross_pow_ele', 'nb_prod_act', 'net_margin',
'pow_max',
       'price_off_peak_var', 'price_peak_var', 'price_mid_pe
ak_var',
       'price_off_peak_fix', 'price_peak_fix', 'price_mid_pe
ak_fix',
       'offpeak_diff_dec_january_energy', 'offpeak_diff_dec_
january_power',
       'month_sub_ended', 'year_sub_ended', 'month_sub_bega
n',
       'channel_MISSING', 'channel_epumfxlbckeskwekxbiuasklx
alciiuu',
       'channel_ewpakwlliwisiwduibdlfmalxowmwpci',
       'channel_foosdfpfkusacimwkcsosbicdxkicaua',
       'channel_lmkebamcaaclubfxadlmueccxoimlema',
       'channel_usilxuppasemubllopkaafesmlibmsdf',
       'origin_kamkkxfxxuwbdslkwifmmcsiusiuosws',
       'origin_ldkssxwpmemidmecebumciepifcamkci',
       'origin_lxidpiddsbxsbosboudacockeimpuepw'],
      dtype='object')
```

In […

```
```

In [8…

```
proba_predictions = modelS.predict_proba(X_tests)
probabilities = proba_predictions[:,1]
```

In [8…

```
X_tests['churn'] = predictionss.tolist()
X_tests['churn_probability'] = probabilities.tolist()
X_tests.to_csv('sampled_data_with_predictions.csv')
```

In […

```
```