In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sb
import warnings
warnings.filterwarnings(action='ignore')
```

In [2]:
```
df1=pd.read_csv('Customer.csv')
```

In [3]:
```
df1.head()
```

Out[3]:

	Unnamed: 0	id	activity_new	campaign_disc_el
0	0	48ada52261e7cf58715202705a0451c9	esoiiifxdlbkcsluxmfuacbdckommixw	NaN
1	1	24011ae4ebbe3035111d65fa7c15bc57	NaN	NaN
2	2	d29c2c54acc38ff3c0614d0a653813dd	NaN	NaN
3	3	764c75f661154dac3a6c254cd082ea7d	NaN	NaN
4	4	bba03439a292a1e166f80264c16191cb	NaN	NaN

5 rows × 34 columns

In [4]:
```
df1.drop(columns=['Unnamed: 0','campaign_disc_ele'],inplace=True)
```

In [5]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16096 entries, 0 to 16095
Data columns (total 32 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   id                      16096 non-null  object 
 1   activity_new            6551 non-null   object 
 2   channel_sales           11878 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  object 
 7   date_end                16094 non-null  object 
 8   date_first_activ        3508 non-null   object 
 9   date_modif_prod         15939 non-null  object 
 10  date_renewal            16056 non-null  object 
 11  forecast_base_bill_ele  3508 non-null   float64
 12  forecast_base_bill_year 3508 non-null   float64
 13  forecast_bill_12m       3508 non-null   float64
 14  forecast_cons           3508 non-null   float64
 15  forecast_cons_12m       16096 non-null  float64
 16  forecast_cons_year      16096 non-null  int64
```

```
 17  forecast_discount_energy  15970 non-null  float64
 18  forecast_meter_rent_12m   16096 non-null  float64
 19  forecast_price_energy_p1  15970 non-null  float64
 20  forecast_price_energy_p2  15970 non-null  float64
 21  forecast_price_pow_p1     15970 non-null  float64
 22  has_gas                   16096 non-null  object
 23  imp_cons                  16096 non-null  float64
 24  margin_gross_pow_ele      16083 non-null  float64
 25  margin_net_pow_ele        16083 non-null  float64
 26  nb_prod_act               16096 non-null  int64
 27  net_margin                16081 non-null  float64
 28  num_years_antig           16096 non-null  int64
 29  origin_up                 16009 non-null  object
 30  pow_max                   16093 non-null  float64
 31  churn                     16096 non-null  int64
dtypes: float64(15), int64(7), object(10)
memory usage: 3.9+ MB
```

Replacing missing values with mean

In [6]:
```
df1_columns=['forecast_base_bill_ele','forecast_base_bill_year','forecast_bill_12m',
             'forecast_price_energy_p1','forecast_price_energy_p2','forecast_price_p
            'net_margin','pow_max']
for i in df1_columns:
    df1[i]=df1[i].fillna(df1[i].mean())
```

In [7]:
```
df1.isnull().sum()
```

Out[7]:
```
id                              0
activity_new                 9545
channel_sales                4218
cons_12m                        0
cons_gas_12m                    0
cons_last_month                 0
date_activ                      0
date_end                        2
date_first_activ            12588
date_modif_prod               157
date_renewal                   40
forecast_base_bill_ele          0
forecast_base_bill_year         0
forecast_bill_12m               0
forecast_cons                   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                      87
pow_max                         0
churn                           0
dtype: int64
```

Transforming categorical variables

In [8]:
```
#has_gas variable
df1['has_gas'].replace({'f':0,'t':1},inplace=True)
df1['has_gas'].unique()
```

Out[8]:
```
array([0, 1], dtype=int64)
```

In [9]:
```
df1['channel_sales']=df1['channel_sales'].fillna('null_values')
```

In [10]:
```
df1['channel_sales']=df1['channel_sales'].astype('category')
len(df1['channel_sales'].unique())
```

Out[10]: 8

In [11]:
```
categories_channel=pd.get_dummies(df1['channel_sales'])
categories_channel.head()
```

Out[11]:

	epumfxlbckeskwekxbiuasklxalciiuu	ewpakwlliwisiwduibdlfmalxowmwpci	fixdbufsefwooaasfcxdxadsi
0	0	0	
1	0	0	
2	0	0	
3	0	0	
4	0	0	

In [12]:
```
categories_channel.drop(columns='null_values',inplace=True)
```

In [13]:
```
df1['origin_up']=df1['origin_up'].fillna('null_values')
```

In [14]:
```
df1['origin_up']=df1['origin_up'].astype('category')
len(df1['origin_up'].unique())
```

Out[14]: 6

In [15]:
```
categories_origin=pd.get_dummies(df1['origin_up'])
categories_origin.head()
```

Out[15]:

	ewxeelcelemmiwuafmddpobolfuxioce	kamkkxfxxuwbdslkwifmmcsiusiuosws	ldkssxwpmemidmecebu
0	0	0	
1	0	0	
2	0	1	
3	0	1	
4	0	1	

In [16]:
```
categories_origin.drop(columns='null_values')
```

Out[16]:

	ewxeelcelemmiwuafmddpobolfuxioce	kamkkxfxxuwbdslkwifmmcsiusiuosws	ldkssxwpmemidm
0	0	0	
1	0	0	
2	0	1	
3	0	1	
4	0	1	
...	...	...	
16091	0	0	
16092	0	0	
16093	0	0	
16094	0	0	
16095	0	0	

16096 rows × 5 columns

In [17]:
```
len(df1['activity_new'].unique())
```

Out[17]: 420

In [18]:
```
df1['activity_new']=df1['activity_new'].fillna('null_values')
```

In [19]:
```
categories_activity=pd.DataFrame({"activity_samples":df1['activity_new'].value_count
categories_activity
```

Out[19]:

	activity_samples
null_values	9545
apdekpcbwosbxepsfxclislboipuxpop	1577
kkklcdamwfafdcfwofuscwfwadblfmce	422
kwuslieomapmswolewpobpplkaooaaew	230
fmwdwsxillemwbbwelxsampiuwwpcdcb	219
...	...
uuxeifdawaobxfxxefkdfxkmsmbfoamf	1
fxocpcbfplipxiokscwiuexkceoucmko	1
dbxlsaldowxpxlxfoueabwbaclmlbuiu	1
xwkaesbkfsacseixxksofpddwfkbobki	1
duidibwdmowiudemasiwabceucckesdw	1

420 rows × 1 columns

In [20]:
```
to_replace=list(categories_activity[categories_activity['activity_samples']<=75].ind
df1['activity_new']=df1['activity_new'].replace(to_replace,"null_values")
```

In [21]:
```
categories_activity=pd.get_dummies(df1['activity_new'])
categories_activity.head()
```

Out[21]:

	apdekpcbwosbxepsfxclislboipuxpop	ckfxocssowaeipxueikxcmaxdmcduxsa	cluecxlameloamldmasudc
0	0	0	
1	0	0	
2	0	0	
3	0	0	
4	0	0	

In [22]:
```
categories_activity.drop(columns='null_values',inplace=True)
```

In [23]:
```
df1=pd.merge(df1,categories_channel,left_index=True,right_index=True)
df1=pd.merge(df1,categories_origin,left_index=True,right_index=True)
df1=pd.merge(df1,categories_activity,left_index=True,right_index=True)
```

Removing skewness in data

In [24]:
```
df1.describe()
```

Out[24]:

	cons_12m	cons_gas_12m	cons_last_month	forecast_base_bill_ele	forecast_base_bill_year
count	1.609600e+04	1.609600e+04	1.609600e+04	16096.000000	16096.000000
mean	1.948044e+05	3.191164e+04	1.946154e+04	335.843857	335.843857
std	6.795151e+05	1.775885e+05	8.235676e+04	303.136819	303.136819
min	-1.252760e+05	-3.037000e+03	-9.138600e+04	-364.940000	-364.940000
25%	5.906250e+03	0.000000e+00	0.000000e+00	335.843857	335.843857
50%	1.533250e+04	0.000000e+00	9.010000e+02	335.843857	335.843857
75%	5.022150e+04	0.000000e+00	4.127000e+03	335.843857	335.843857
max	1.609711e+07	4.188440e+06	4.538720e+06	12566.080000	12566.080000

8 rows × 45 columns

In [25]:
```
from scipy.stats import skew
for col in df1_columns:
    print(col)
    print(skew(df1[col]))
```

```
forecast_base_bill_ele
```

```
14.225129736902673
forecast_base_bill_year
14.225129736902673
forecast_bill_12m
10.635845725001651
forecast_cons
14.80491423522851
forecast_discount_energy
5.123134537067522
forecast_price_energy_p1
0.2257417234042362
forecast_price_energy_p2
-0.12450138011516816
forecast_price_pow_p1
-2.532755513603452
margin_gross_pow_ele
1.0489029493687139
margin_net_pow_ele
-3.126399641289942
net_margin
21.32139691727201
pow_max
6.941839188425517
```

In [26]:

```
df1.loc[df1.cons_12m<0,"cons_12m"]=np.nan
df1.loc[df1.cons_gas_12m<0,"cons_gas_12m"]=np.nan
df1.loc[df1.cons_last_month<0,"cons_last_month"]=np.nan
df1.loc[df1.forecast_base_bill_ele<0,"forecast_base_bill_ele"]=np.nan
df1.loc[df1.forecast_base_bill_year<0,"forecast_base_bill_year"]=np.nan
df1.loc[df1.forecast_bill_12m<0,"forecast_bill_12m"]=np.nan
df1.loc[df1.forecast_cons<0,"forecast_cons"]=np.nan
df1.loc[df1.forecast_discount_energy<0,"forecast_discount_energy"]=np.nan
df1.loc[df1.forecast_price_energy_p1<0,"forecast_price_energy_p1"]=np.nan
df1.loc[df1.forecast_price_energy_p2<0,"forecast_price_energy_p2"]=np.nan
df1.loc[df1.forecast_price_pow_p1<0,"forecast_price_pow_p1"]=np.nan
```

In [27]:

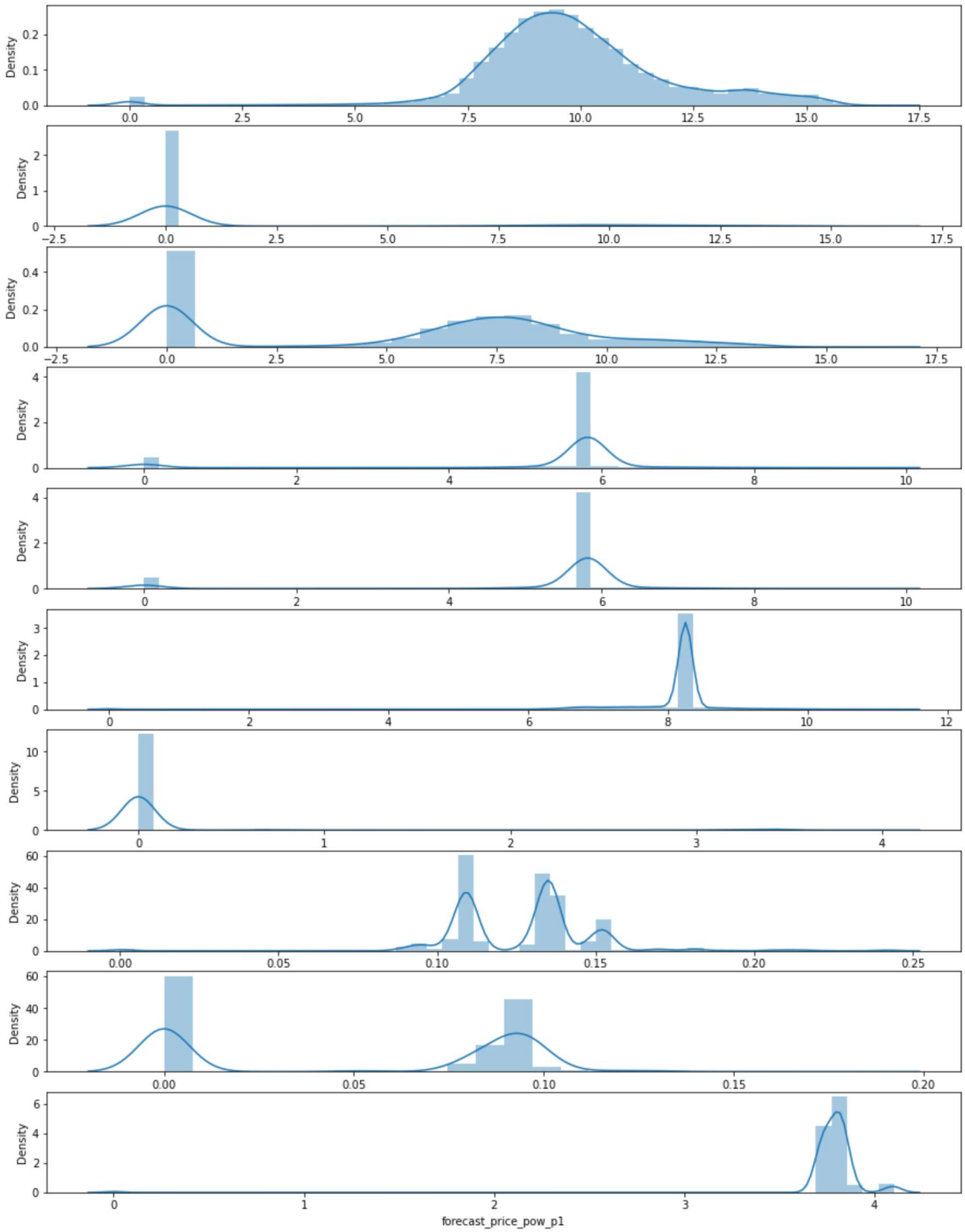
```
df1['cons_12m']=np.log(df1['cons_12m']+1)
df1['cons_gas_12m']=np.log(df1['cons_gas_12m']+1)
df1['cons_last_month']=np.log(df1['cons_last_month']+1)
df1['forecast_base_bill_ele']=np.log(df1['forecast_base_bill_ele']+1)
df1['forecast_base_bill_year']=np.log(df1['forecast_base_bill_year']+1)
df1['forecast_bill_12m']=np.log(df1['forecast_bill_12m']+1)
df1['forecast_discount_energy']=np.log(df1['forecast_discount_energy']+1)
df1['forecast_price_energy_p1']=np.log(df1['forecast_price_energy_p1']+1)
df1['forecast_price_energy_p2']=np.log(df1['forecast_price_energy_p2']+1)
df1['forecast_price_pow_p1']=np.log(df1['forecast_price_pow_p1']+1)
```

In [29]:

```
fig, axs=plt.subplots(nrows=10,figsize=(15,20))
sb.distplot((df1['cons_12m']).dropna(),ax=axs[0])
sb.distplot((df1['cons_gas_12m']).dropna(),ax=axs[1])
sb.distplot((df1['cons_last_month']).dropna(),ax=axs[2])
sb.distplot((df1['forecast_base_bill_ele']).dropna(),ax=axs[3])
sb.distplot((df1['forecast_base_bill_year']).dropna(),ax=axs[4])
sb.distplot((df1['forecast_bill_12m']).dropna(),ax=axs[5])
sb.distplot((df1['forecast_discount_energy']).dropna(),ax=axs[6])
sb.distplot((df1['forecast_price_energy_p1']).dropna(),ax=axs[7])
sb.distplot((df1['forecast_price_energy_p2']).dropna(),ax=axs[8])
sb.distplot((df1['forecast_price_pow_p1']).dropna(),ax=axs[9])
```

Out[29]:

```
<AxesSubplot:xlabel='forecast_price_pow_p1', ylabel='Density'>
```

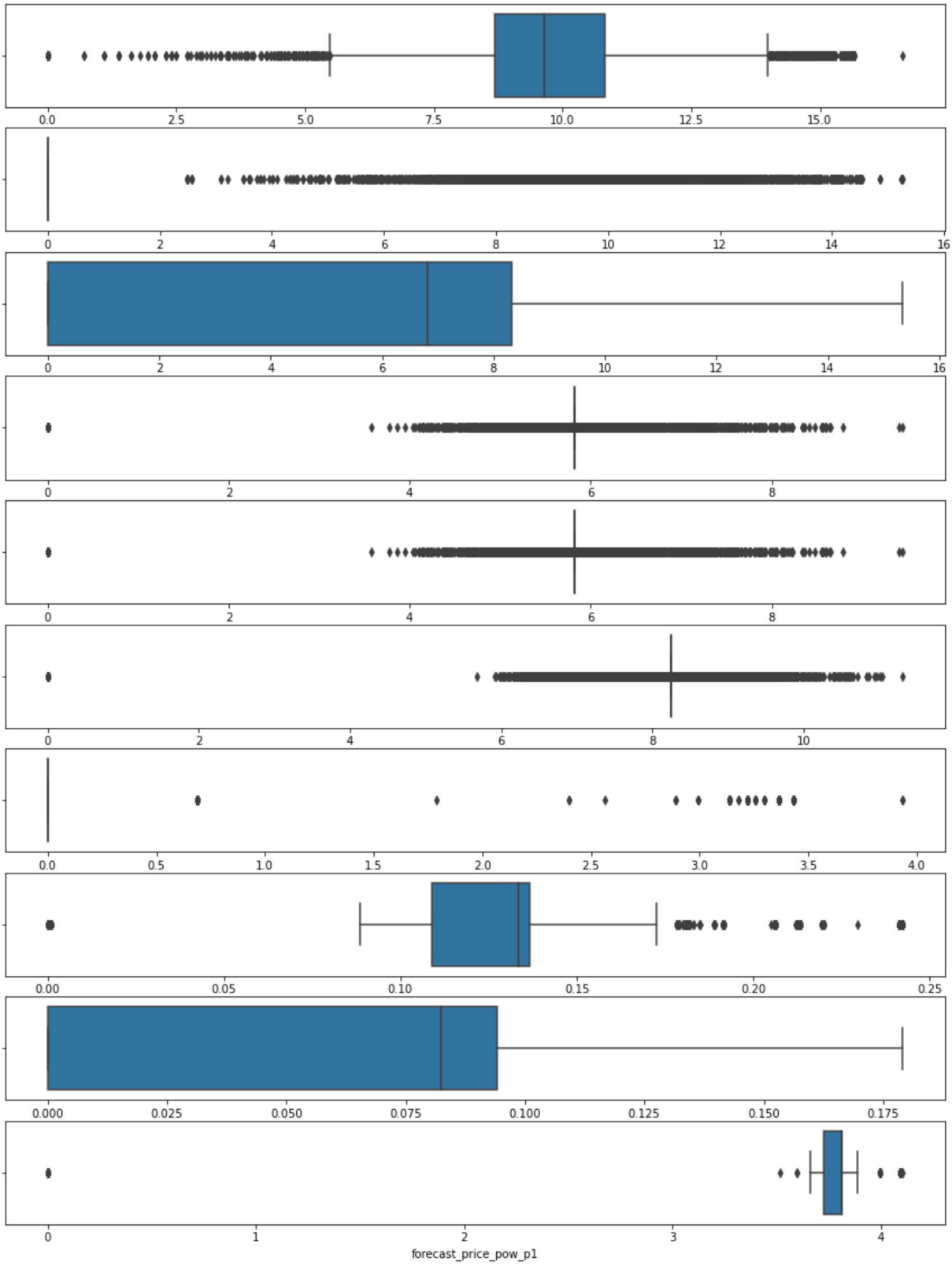


Outlier cleaning

In [33]:

```
fig, axs=plt.subplots(nrows=10,figsize=(15,20))
sb.boxplot(df1['cons_12m'].dropna(),ax=axs[0])
sb.boxplot(df1['cons_gas_12m'].dropna(),ax=axs[1])
sb.boxplot(df1['cons_last_month'].dropna(),ax=axs[2])
sb.boxplot(df1['forecast_base_bill_ele'].dropna(),ax=axs[3])
sb.boxplot(df1['forecast_base_bill_year'].dropna(),ax=axs[4])
sb.boxplot(df1['forecast_bill_12m'].dropna(),ax=axs[5])
sb.boxplot(df1['forecast_discount_energy'].dropna(),ax=axs[6])
sb.boxplot(df1['forecast_price_energy_p1'].dropna(),ax=axs[7])
sb.boxplot(df1['forecast_price_energy_p2'].dropna(),ax=axs[8])
```

```
sb.boxplot(df1['forecast_price_pow_p1'].dropna(),ax=axs[9])
plt.show()
```



In [38]:

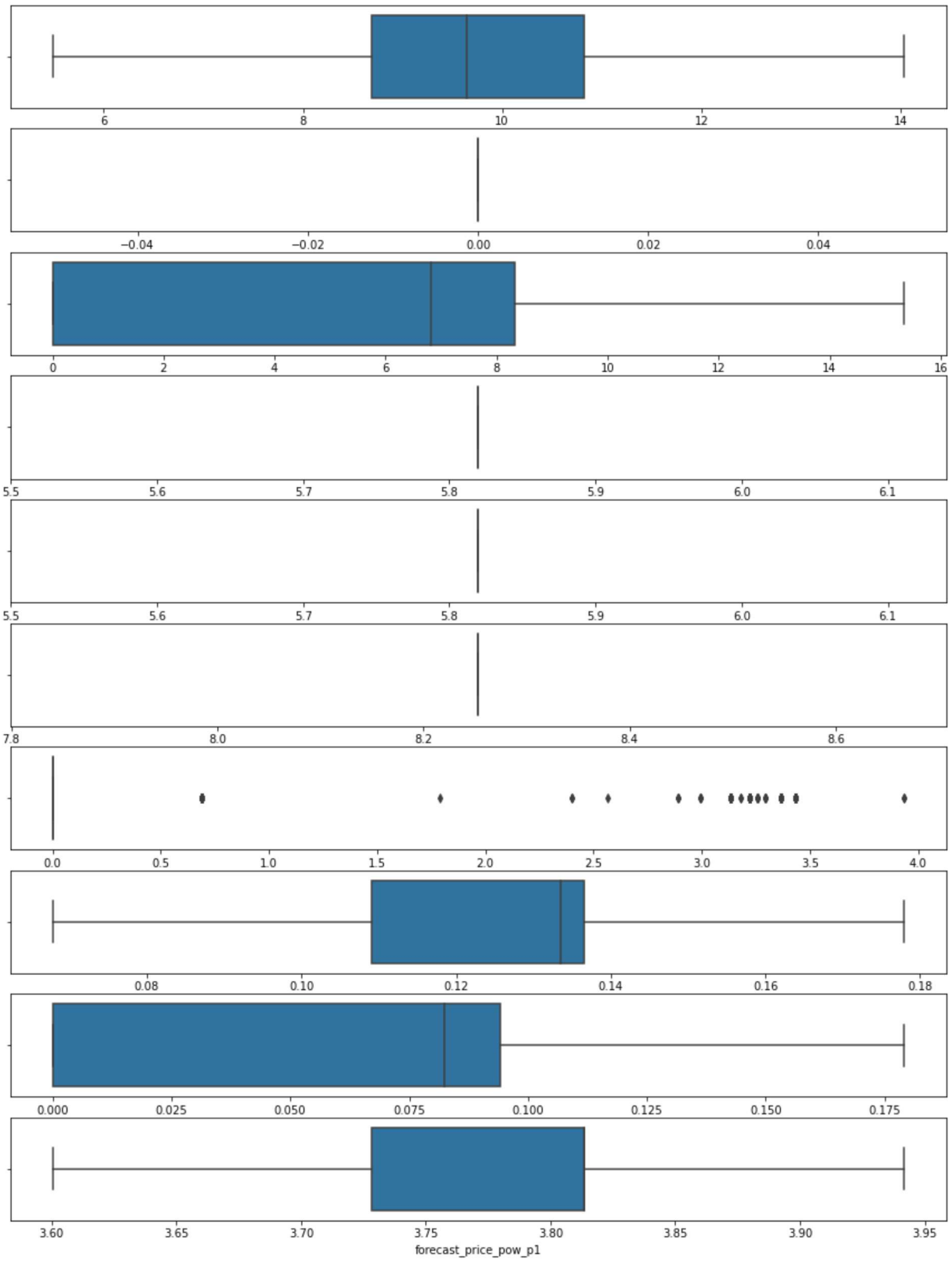
```
columns=['cons_12m','cons_gas_12m','cons_last_month','forecast_base_bill_ele','forec
        'forecast_cons','forecast_cons_12m','forecast_cons_year','forecast_price_ene
        'forecast_price_pow_p1','imp_cons','margin_gross_pow_ele','margin_net_pow_el
        'net_margin','num_years_antig','pow_max']

for i in columns:
    Q1=df1[i].quantile(.25)
    Q3=df1[i].quantile(.75)
    IQR=Q3-Q1
    lower_limit=Q1-1.5*(IQR)
```

```
    upper_limit=Q3+1.5*(IQR)
    def limit_imputer(value):
        if value>upper_limit:
            return upper_limit
        if value<lower_limit:
            return lower_limit
        else:
            return value
    df1[i]=df1[i].apply(limit_imputer)
```

In [39]:

```
fig, axs=plt.subplots(nrows=10,figsize=(15,20))
sb.boxplot(df1['cons_12m'].dropna(),ax=axs[0])
sb.boxplot(df1['cons_gas_12m'].dropna(),ax=axs[1])
sb.boxplot(df1['cons_last_month'].dropna(),ax=axs[2])
sb.boxplot(df1['forecast_base_bill_ele'].dropna(),ax=axs[3])
sb.boxplot(df1['forecast_base_bill_year'].dropna(),ax=axs[4])
sb.boxplot(df1['forecast_bill_12m'].dropna(),ax=axs[5])
sb.boxplot(df1['forecast_discount_energy'].dropna(),ax=axs[6])
sb.boxplot(df1['forecast_price_energy_p1'].dropna(),ax=axs[7])
sb.boxplot(df1['forecast_price_energy_p2'].dropna(),ax=axs[8])
sb.boxplot(df1['forecast_price_pow_p1'].dropna(),ax=axs[9])
plt.show()
```


6
8
10
12
14
−0.04
−0.02
0.00
0.02
0.04
0
2
4
6
8
10
12
14
16
5.5
5.6
5.7
5.8
5.9
6.0
6.1
5.5
5.6
5.7
5.8
5.9
6.0
6.1
7.8
8.0
8.2
8.4
8.6
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
0.08
0.10
0.12
0.14
0.16
0.18
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
3.60
3.65
3.70
3.75
3.80
3.85
3.90
3.95
forecast_price_pow_p1