In [1]:
```python
try:
    from IPython import get_ipython
    get_ipython().magic('clear')
    get_ipython().magic('reset -f')
except:
    pass
```

In [2]:
```python
#Importing all the required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import missingno as msno
from scipy.stats import zscore as zscore
#To ignore the warnings
import warnings as wg
wg.filterwarnings("ignore")
```

In [3]:
```python
# Set plot style
sns.set(color_codes=True)
```

In [4]:
```python
# Set maximum number of columns to be displayed
pd.set_option('display.max_columns', 100)
```

In [5]:
```python
#Read the dataset file of train data
train_data=pd.read_csv("ml_case_training_data.csv")
#Read the dataset file of history data
history_data=pd.read_csv("ml_case_training_hist_data.csv")
#Read the dataset file of churn data
churn_data=pd.read_csv("ml_case_training_output.csv")
```

In [6]:
```python
train = pd.merge(train_data, churn_data, on="id")
```

In [7]:
```python
train.head()
```

Out[7]:

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

In [8]:
```python
pd.DataFrame({"Data type":train.dtypes})
```

Out[8]:

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

In [9]:
```python
pd.DataFrame({"Data type":history_data.dtypes})
```

Out[9]:

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

In [10]:
```python
train.describe()
```

Out[10]:

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

In [11]:
```python
train["campaign_disc_ele"].isnull().values.all()
```

Out[11]:

```
True
```

In [12]:
```python
history_data.describe()
```

Out[12]:

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

In [13]:
```python
pd.DataFrame({"Missing values (%)": train.isnull().sum()/len(train.index)*100})
```

Out[13]:

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

In [14]:
```python
pd.DataFrame({"Missing values (%)": history_data.isnull().sum()/len(history_data.index)*100})
```

Out[14]:

	Missing values (%)
id	0.000000
price_date	0.000000
price_p1_var	0.704138
price_p2_var	0.704138
price_p3_var	0.704138
price_p1_fix	0.704138
price_p2_fix	0.704138
price_p3_fix	0.704138

In [15]:
```python
train[train['date_end'].isnull()]
```

Out[15]:

	id	activity_new	campaign_disc_ele	channel_sales	cons_12m	cons_gas_12m	cons_last_month	date_activ	date_end	date_
4452	f7aed20b79f48e9e87687dc74fc6a353	ilkfsaapsxpkcpswblldfmpamwelpxi	NaN	foosdfpfkusacimwkcsosbicdxkicaua	107905	0	0	2013-06-19	NaN	
12880	ec6c8b46866d76e99623ddcf9d334835	mloxfbllfoxllsffauklsewwfcfdlls	NaN	NaN	18954	0	1596	2010-09-06	NaN	

In [16]:
```python
train.drop(['campaign_disc_ele','date_first_activ','forecast_base_bill_ele','forecast_cons','forecast_bill_12m','forecast_base_bill_year'],inplace=True,axis=1
```

In [17]:
```python
train.isnull().sum()
```

Out[17]:

```
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_modif_prod             157
date_renewal                 40
forecast_cons_12m             0
forecast_cons_year            0
forecast_discount_energy    126
forecast_meter_rent_12m       0
forecast_price_energy_p1    126
forecast_price_energy_p2    126
forecast_price_pow_p1       126
has_gas                       0
imp_cons                      0
margin_gross_pow_ele         13
margin_net_pow_ele           13
nb_prod_act                   0
net_margin                   15
num_years_antig               0
origin_up                    87
pow_max                       3
churn                         0
dtype: int64
```

In [18]:
```python
df=train.head()
```

In [19]:
```python
def impute_nan(df,variable,median):
    df[variable]=df[variable].fillna(median)
```

In [20]:
```python
median=df.net_margin.median()
```

In [21]:
```python
median
```

Out[21]:

```
25.46
```

In [22]:
```python
impute_nan(df,'forecast_discount_energy',median)
impute_nan(df,'forecast_price_energy_p1',median)
impute_nan(df,'forecast_price_energy_p2',median)
impute_nan(df,'forecast_price_pow_p1',median)
impute_nan(df,'margin_gross_pow_ele',median)
impute_nan(df,'margin_net_pow_ele',median)
impute_nan(df,'net_margin',median)
impute_nan(df,'pow_max',median)
```

In [23]:
```python
def impute_nan(df,variable):
    most_frequent_category=df[variable].mode()[0]
    df[variable].fillna(most_frequent_category,inplace=True)
```

In [24]:
```python
for feature in ['activity_new','channel_sales','date_modif_prod']:
    impute_nan(df,feature)
```

In [25]:
```python
df.isnull().sum()
```

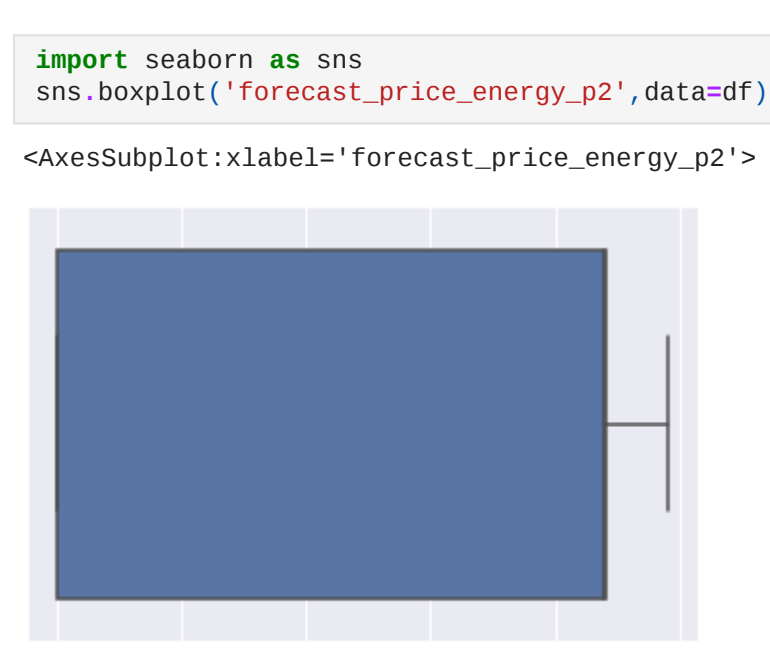
Out[25]:

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

In [26]:
```python
import matplotlib.pyplot as plt
%matplotlib inline
```

In [27]:
```python
df.forecast_discount_energy.hist(bins=50)
```

Out[27]: <AxesSubplot:>

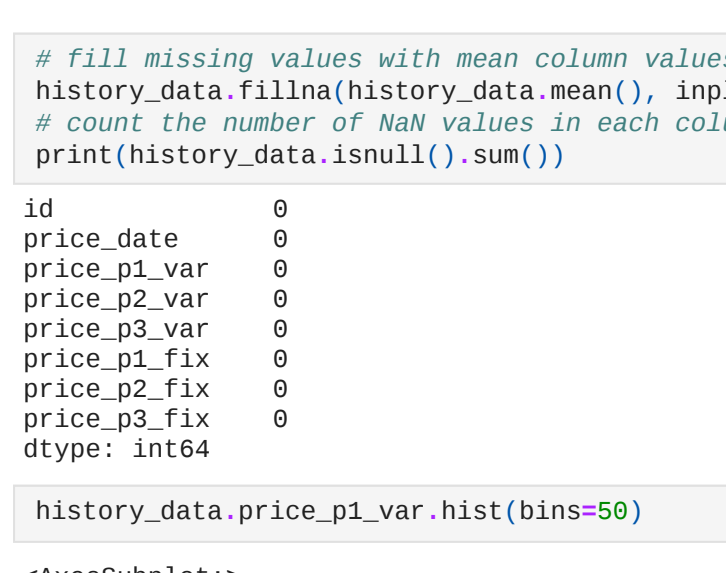


In [28]:
```python
import seaborn as sns
sns.boxplot('forecast_price_energy_p1',data=df)
```

Out[28]: <AxesSubplot:xlabel='forecast_price_energy_p1'>

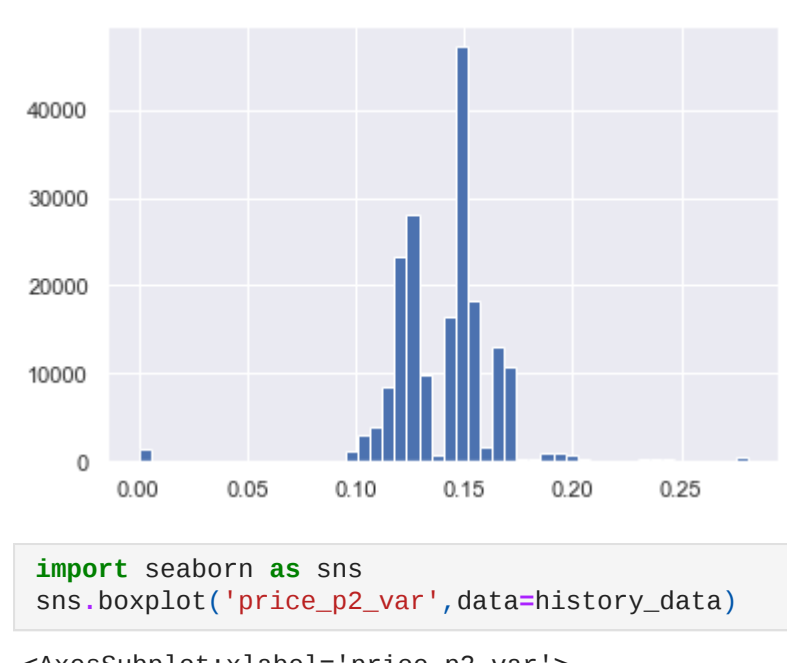
In [29]:
```python
import seaborn as sns
sns.boxplot('forecast_price_energy_p2',data=df)
```

Out[29]: <AxesSubplot:xlabel='forecast_price_energy_p2'>



In [30]:
```python
df.forecast_price_energy_p2.hist(bins=50)
```

Out[30]: <AxesSubplot:>

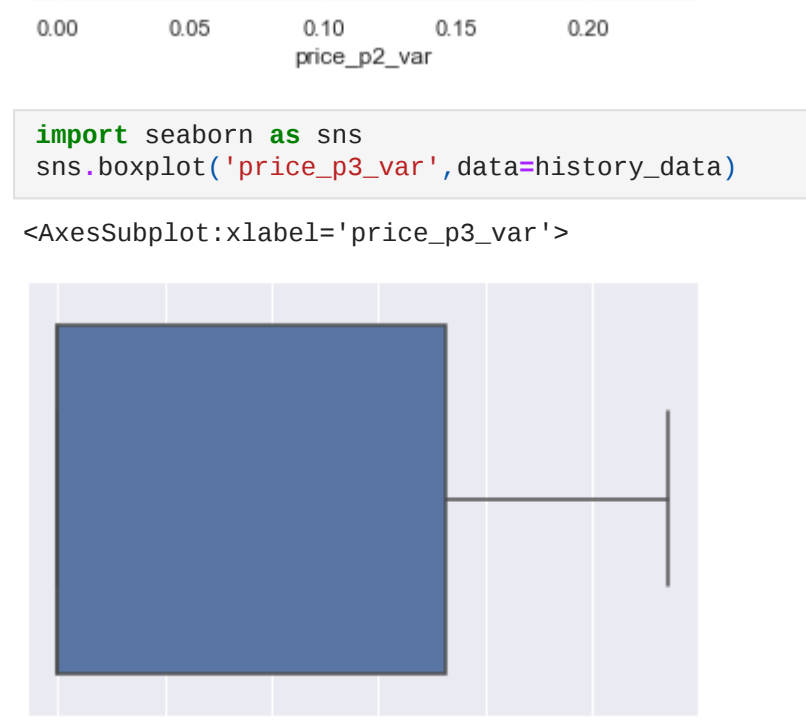


In [31]:
```python
# fill missing values with mean column values
history_data.fillna(history_data.mean(), inplace=True)
# count the number of NaN values in each column
print(history_data.isnull().sum())
```

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

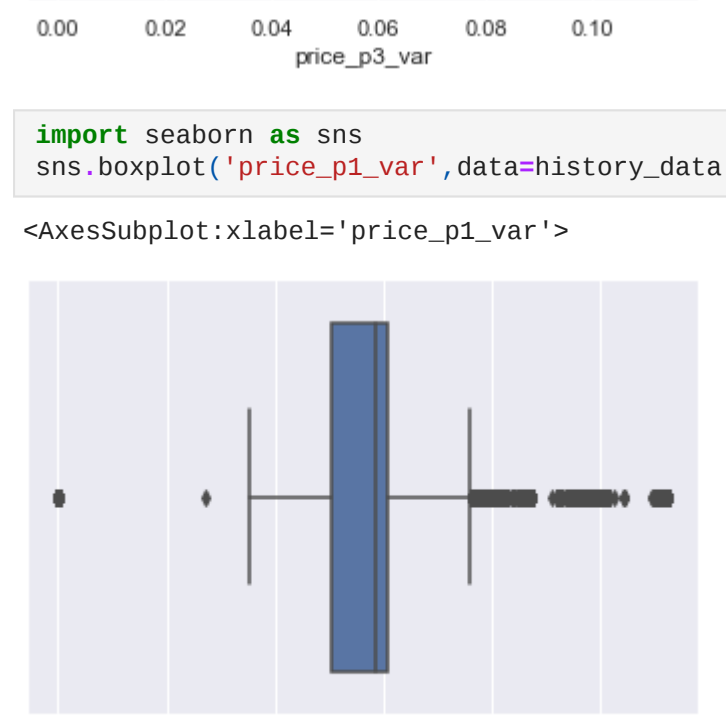
In [32]:
```python
history_data.price_p1_var.hist(bins=50)
```

Out[32]: <AxesSubplot:>



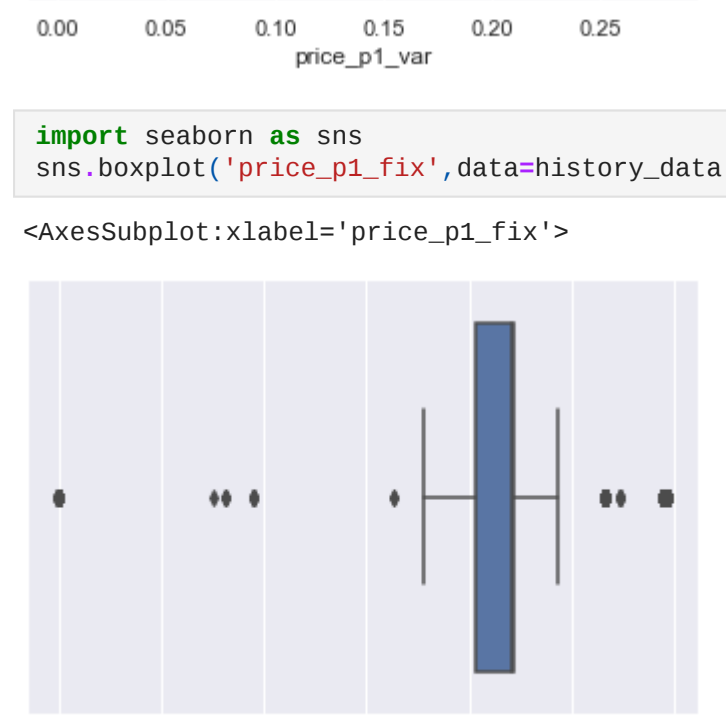
In [33]:
```python
import seaborn as sns
sns.boxplot('price_p2_var',data=history_data)
```

Out[33]: <AxesSubplot:xlabel='price_p2_var'>



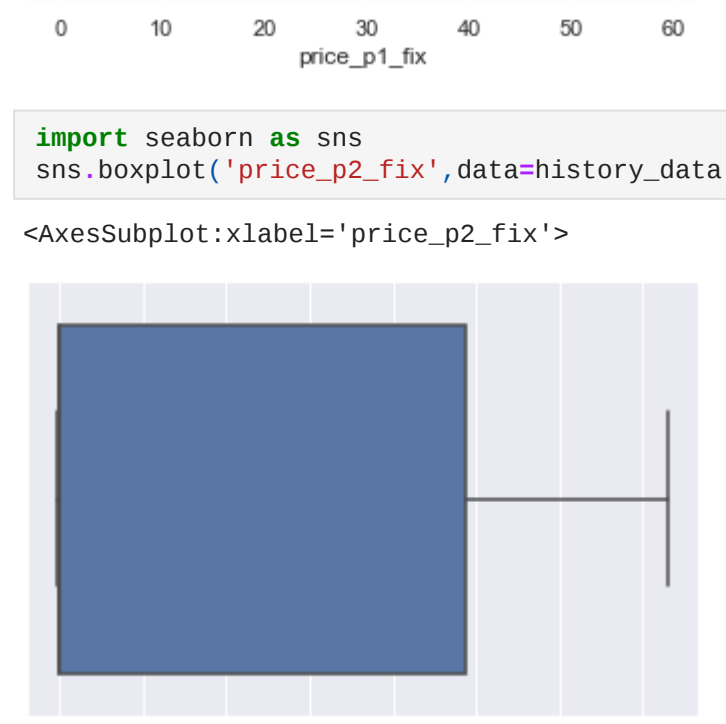
In [34]:
```python
import seaborn as sns
sns.boxplot('price_p3_var',data=history_data)
```

Out[34]: <AxesSubplot:xlabel='price_p3_var'>



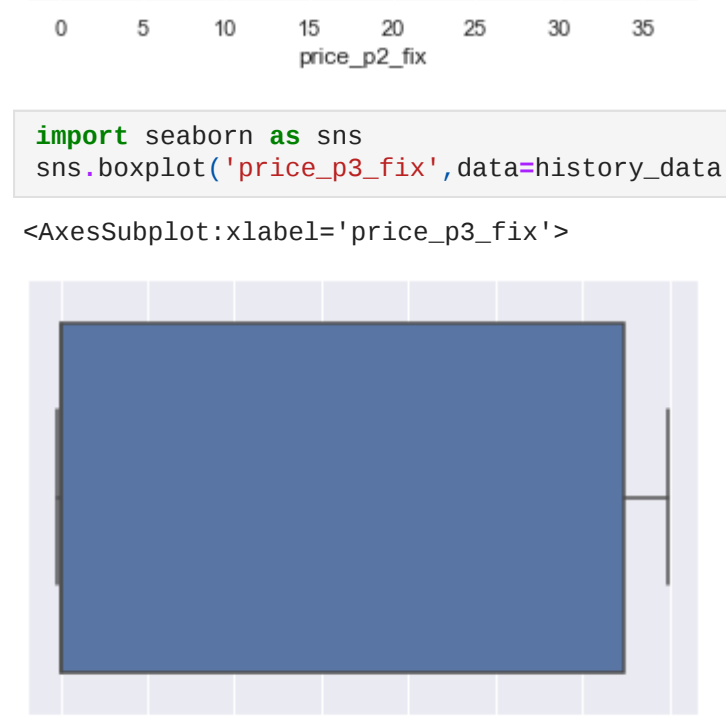
In [35]:
```python
import seaborn as sns
sns.boxplot('price_p1_var',data=history_data)
```

Out[35]: <AxesSubplot:xlabel='price_p1_var'>



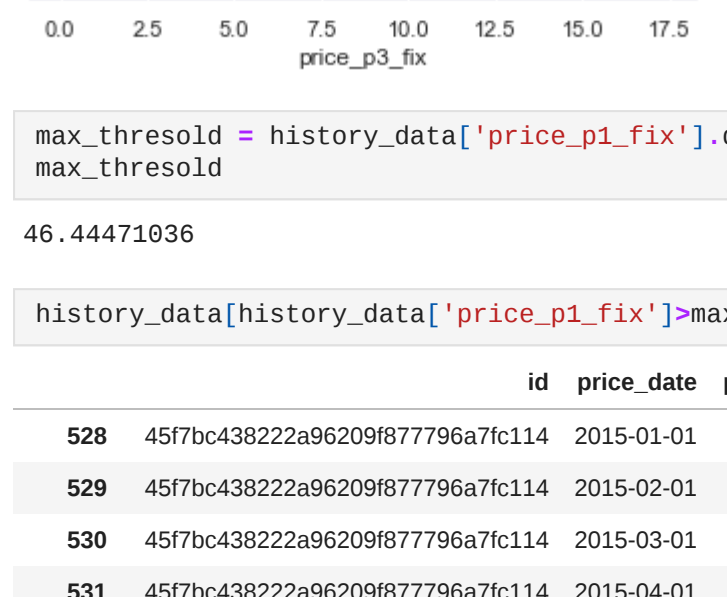
In [36]:
```python
import seaborn as sns
sns.boxplot('price_p1_fix',data=history_data)
```

Out[36]: <AxesSubplot:xlabel='price_p1_fix'>



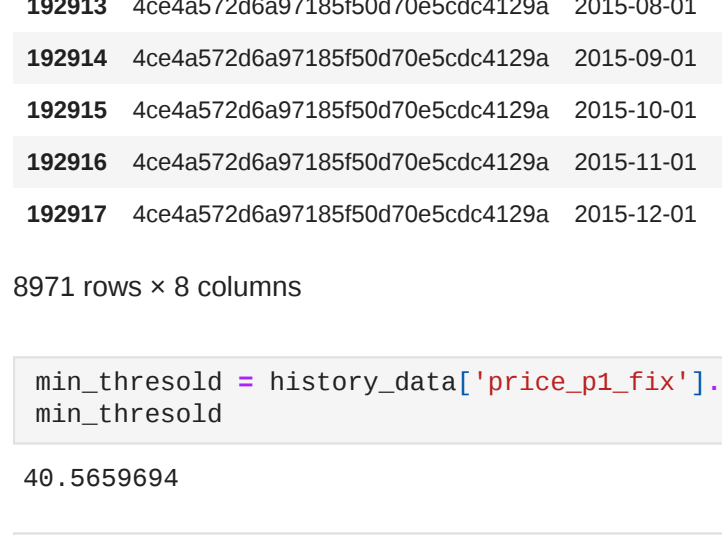
In [37]:
```python
import seaborn as sns
sns.boxplot('price_p2_fix',data=history_data)
```

Out[37]: <AxesSubplot:xlabel='price_p2_fix'>



In [38]:
```python
import seaborn as sns
sns.boxplot('price_p3_fix',data=history_data)
```

Out[38]: <AxesSubplot:xlabel='price_p3_fix'>



In [40]:
```python
max_thresold = history_data['price_p1_fix'].quantile(0.95)
max_thresold
```

Out[40]:

```
46.44471036
```

In [42]:
```python
history_data[history_data['price_p1_fix']>max_thresold]
```

Out[42]:

	id	price_date	price_p1_var	price_p2_var	price_p3_var	price_p1_fix	price_p2_fix	price_p3_fix
528	45f7bc438222a96209f877796a7fc114	2015-01-01	0.107491	0.094600	0.068317	59.173468	36.490689	8.367731
529	45f7bc438222a96209f877796a7fc114	2015-02-01	0.107491	0.094600	0.068317	59.173468	36.490689	8.367731
530	45f7bc438222a96209f877796a7fc114	2015-03-01	0.107491	0.094600	0.068317	59.173468	36.490689	8.367731
531	45f7bc438222a96209f877796a7fc114	2015-04-01	0.107491	0.094600	0.068317	59.173468	36.490689	8.367731
532	45f7bc438222a96209f877796a7fc114	2015-05-01	0.107491	0.094600	0.068317	59.173468	36.490689	8.367731
...	...	...	...	...	...	...	...	...
192913	4ce4a572d6a97185f50d70e5cdc4129a	2015-08-01	0.104531	0.094642	0.071661	58.936780	36.344721	8.334263
192914	4ce4a572d6a97185f50d70e5cdc4129a	2015-09-01	0.100262	0.090362	0.067368	58.936780	36.344721	8.334263
192915	4ce4a572d6a97185f50d70e5cdc4129a	2015-10-01	0.100262	0.090362	0.067368	58.936780	36.344721	8.334263
192916	4ce4a572d6a97185f50d70e5cdc4129a	2015-11-01	0.098414	0.089947	0.066059	59.173468	36.490689	8.367731
192917	4ce4a572d6a97185f50d70e5cdc4129a	2015-12-01	0.098414	0.089947	0.066059	59.173468	36.490689	8.367731

8971 rows × 8 columns

In [44]:
```python
min_thresold = history_data['price_p1_fix'].quantile(0.05)
min_thresold
```

Out[44]:

```
40.5659694
```

In [45]:
```python
history_data[history_data['price_p1_fix']<min_thresold]
```

Out[45]:

	id	price_date	price_p1_var	price_p2_var	price_p3_var	price_p1_fix	price_p2_fix	price_p3_fix
12	31f2ce549924679a3cbb2d128ae9ea43	2015-01-01	0.125976	0.103395	0.071536	40.565969	24.339581	16.226389
13	31f2ce549924679a3cbb2d128ae9ea43	2015-02-01	0.125976	0.103395	0.071536	40.565969	24.339581	16.226389
14	31f2ce549924679a3cbb2d128ae9ea43	2015-03-01	0.125976	0.103395	0.071536	40.565969	24.339581	16.226389
15	31f2ce549924679a3cbb2d128ae9ea43	2015-04-01	0.125976	0.103395	0.071536	40.565969	24.339581	16.226389
16	31f2ce549924679a3cbb2d128ae9ea43	2015-05-01	0.124815	0.102234	0.070375	40.565969	24.339581	16.226389
...	...	...	...	...	...	...	...	...
192920	feefbb8a6b2491c45d6fbd5ab39dcfa3	2015-03-01	0.122976	0.100852	0.067111	40.565969	24.339581	16.226389
192990	16f51cdc2baa19af0b940ee1b3dd17d5	2015-01-01	0.129444	0.106863	0.075004	40.565969	24.339581	16.226389
192991	16f51cdc2baa19af0b940ee1b3dd17d5	2015-02-01	0.129444	0.106863	0.075004	40.565969	24.339581	16.226389
192992	16f51cdc2baa19af0b940ee1b3dd17d5	2015-03-01	0.129444	0.106863	0.075004	40.565969	24.339581	16.226389
192993	16f51cdc2baa19af0b940ee1b3dd17d5	2015-04-01	0.129444	0.106863	0.075004	40.565969	24.339581	16.226389

24125 rows × 8 columns

In [46]:
```python
history_data[(history_data['price_p1_fix']<max_thresold) & (history_data['price_p1_fix']>min_thresold)]
```

Out[46]:

	id	price_date	price_p1_var	price_p2_var	price_p3_var	price_p1_fix	price_p2_fix	price_p3_fix
0	038af19179925da21a25619c5a24b745	2015-01-01	0.151367	0.000000	0.000000	44.266931	0.00000	0.000000
1	038af19179925da21a25619c5a24b745	2015-02-01	0.151367	0.000000	0.000000	44.266931	0.00000	0.000000
2	038af19179925da21a25619c5a24b745	2015-03-01	0.151367	0.000000	0.000000	44.266931	0.00000	0.000000
3	038af19179925da21a25619c5a24b745	2015-04-01	0.149626	0.000000	0.000000	44.266931	0.00000	0.000000
4	038af19179925da21a25619c5a24b745	2015-05-01	0.149626	0.000000	0.000000	44.266931	0.00000	0.000000
...	...	...	...	...	...	...	...	...
192997	16f51cdc2baa19af0b940ee1b3dd17d5	2015-08-01	0.119916	0.102232	0.076257	40.728885	24.43733	16.291555
192998	16f51cdc2baa19af0b940ee1b3dd17d5	2015-09-01	0.119916	0.102232	0.076257	40.728885	24.43733	16.291555
192999	16f51cdc2baa19af0b940ee1b3dd17d5	2015-10-01	0.119916	0.102232	0.076257	40.728885	24.43733	16.291555
193000	16f51cdc2baa19af0b940ee1b3dd17d5	2015-11-01	0.119916	0.102232	0.076257	40.728885	24.43733	16.291555
193001	16f51cdc2baa19af0b940ee1b3dd17d5	2015-12-01	0.119916	0.102232	0.076257	40.728885	24.43733	16.291555

159201 rows × 8 columns