PROJECT–Computer hardware

February 1, 2023

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

[133]:
```
df=pd.read_table('https://archive.ics.uci.edu/ml/machine-learning-databases/
↪cpu-performance/machine.
↪data',sep=',',header=None,names=['Vendors','Model','MYCT','MMIN','MMAX','CACH','CHMIN','CHM
```

[134]:
```
df.head()
```

[134]:

	Vendors	Model	MYCT	MMIN	MMAX	CACH	CHMIN	CHMAX	PRP	ERP
0	adviser	32/60	125	256	6000	256	16	128	198	199
1	amdahl	470v/7	29	8000	32000	32	8	32	269	253
2	amdahl	470v/7a	29	8000	32000	32	8	32	220	253
3	amdahl	470v/7b	29	8000	32000	32	8	32	172	253
4	amdahl	470v/7c	29	8000	16000	32	8	16	132	132

[148]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209 entries, 0 to 208
Data columns (total 10 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   Vendors  209 non-null    object
 1   Model    209 non-null    object
 2   MYCT     209 non-null    int64
 3   MMIN     209 non-null    int64
 4   MMAX     209 non-null    int64
 5   CACH     209 non-null    int64
 6   CHMIN    209 non-null    int64
 7   CHMAX    209 non-null    int64
 8   PRP      209 non-null    int64
 9   ERP      209 non-null    int64
dtypes: int64(8), object(2)
```

```
memory usage: 16.5+ KB
```

[149]:
```
df.isna().sum()
```

[149]:
```
Vendors    0
Model      0
MYCT       0
MMIN       0
MMAX       0
CACH       0
CHMIN      0
CHMAX      0
PRP        0
ERP        0
dtype: int64
```

1 EDA

[150]:
```
df.head()
```

[150]:

	Vendors	Model	MYCT	MMIN	MMAX	CACH	CHMIN	CHMAX	PRP	ERP
0	adviser	32/60	125	256	6000	256	16	128	198	199
1	amdahl	470v/7	29	8000	32000	32	8	32	269	253
2	amdahl	470v/7a	29	8000	32000	32	8	32	220	253
3	amdahl	470v/7b	29	8000	32000	32	8	32	172	253
4	amdahl	470v/7c	29	8000	16000	32	8	16	132	132

[151]:
```
df.isna().sum()
```

[151]:
```
Vendors    0
Model      0
MYCT       0
MMIN       0
MMAX       0
CACH       0
CHMIN      0
CHMAX      0
PRP        0
ERP        0
dtype: int64
```

[145]:
```
df['ERP'].unique()
```

[145]:
```
array([ 199,  253,  132,  290,  381,  749, 1238,   23,   24,   70,  117,
         15,   64,   29,   22,  124,   35,   39,   40,   45,   28,   21,
         27,  102,   74,  138,  136,   44,   30,   41,   54,   18,   36,
         38,   34,   19,   72,   56,   42,   75,  113,  157,   20,   33,
```

```
 47,   25,   52,   50,   53,   73,   32,  175,   57,  181,   82,
171,  361,  350,  220,   17,   26,   31,   76,   59,   65,  101,
116,  128,   37,   46,   80,   88,   86,   95,  107,  119,  120,
 48,  126,  266,  270,  426,  151,  267,  603,   62,   78,  142,
281,  190,   67,   43,   99,   81,  149,  183,  275,  382,  182,
227,  341,  360,  919,  978], dtype=int64)
```

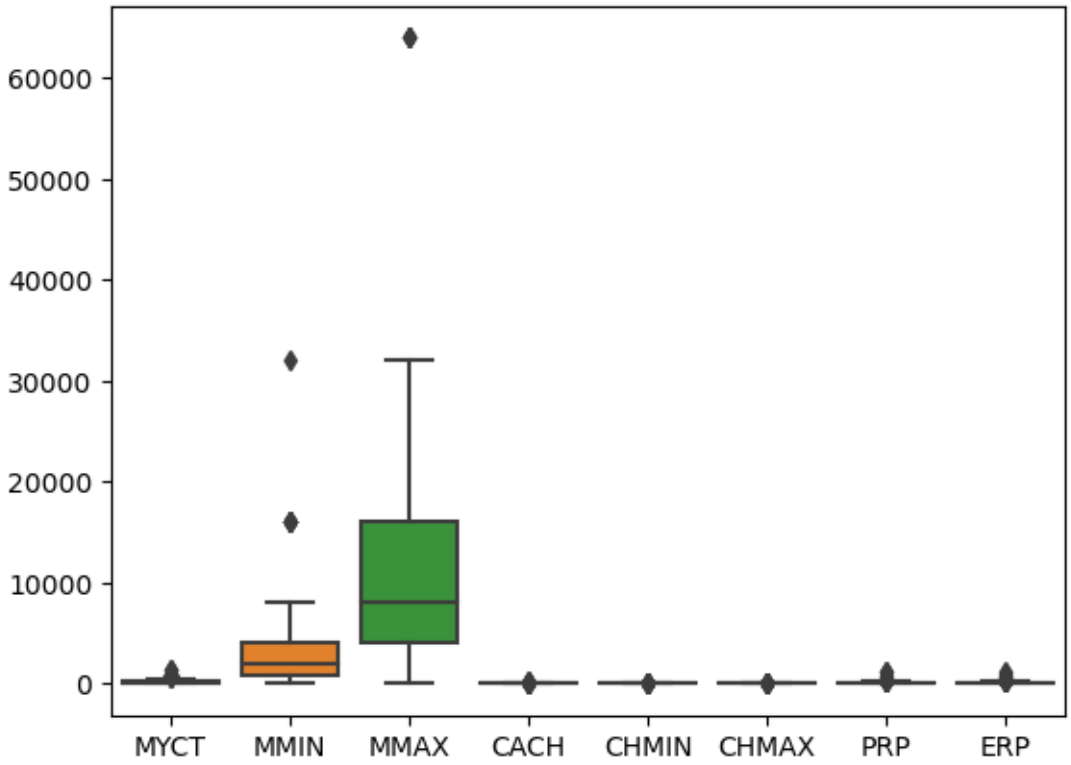
[162]:
```
df.replace([np.inf, -np.inf], np.nan, inplace=True)
```

[147]:
```
from sklearn.impute import SimpleImputer
```

[167]:
```
si=SimpleImputer(missing_values=np.nan,strategy='mean')
df[['CHMAX']]=si.fit_transform(df[['CHMAX']])
```

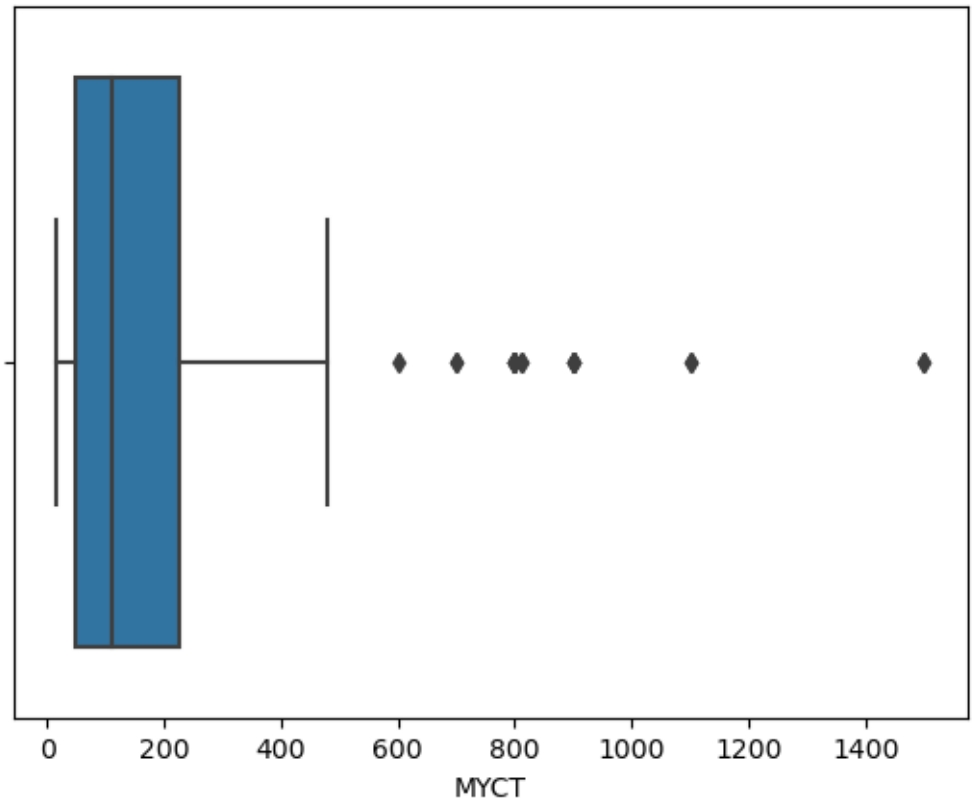
[152]:
```
sns.boxplot(data=df)
```

[152]: <AxesSubplot:>



[153]:
```
sns.boxplot(x='MYCT',data=df)
```

[153]: <AxesSubplot:xlabel='MYCT'>



[154]:
```
colname=df.select_dtypes(['int64']).columns
```

[155]:
```
df[colname]
```

[155]:

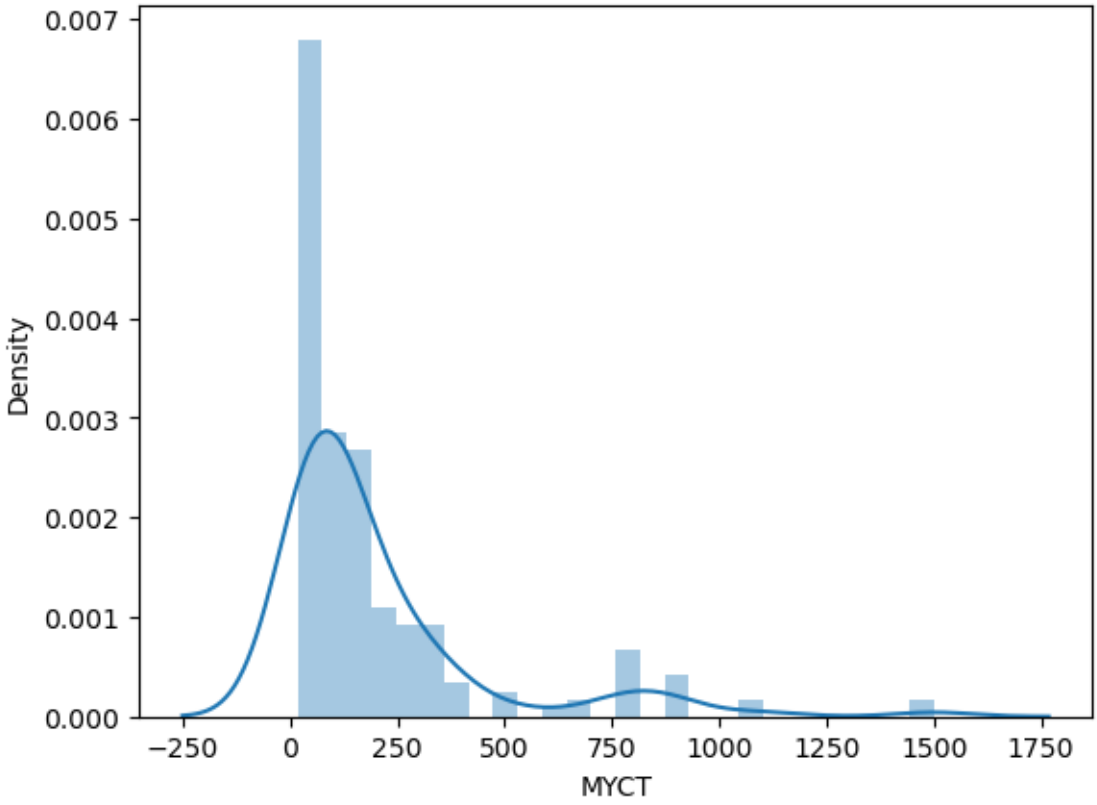
	MYCT	MMIN	MMAX	CACH	CHMIN	CHMAX	PRP	ERP
0	125	256	6000	256	16	128	198	199
1	29	8000	32000	32	8	32	269	253
2	29	8000	32000	32	8	32	220	253
3	29	8000	32000	32	8	32	172	253
4	29	8000	16000	32	8	16	132	132
..	…	…	…	…	…	…	…	…
204	124	1000	8000	0	1	8	42	37
205	98	1000	8000	32	2	8	46	50
206	125	2000	8000	0	2	14	52	41
207	480	512	8000	32	0	0	67	47
208	480	1000	4000	0	0	0	45	25

[209 rows x 8 columns]

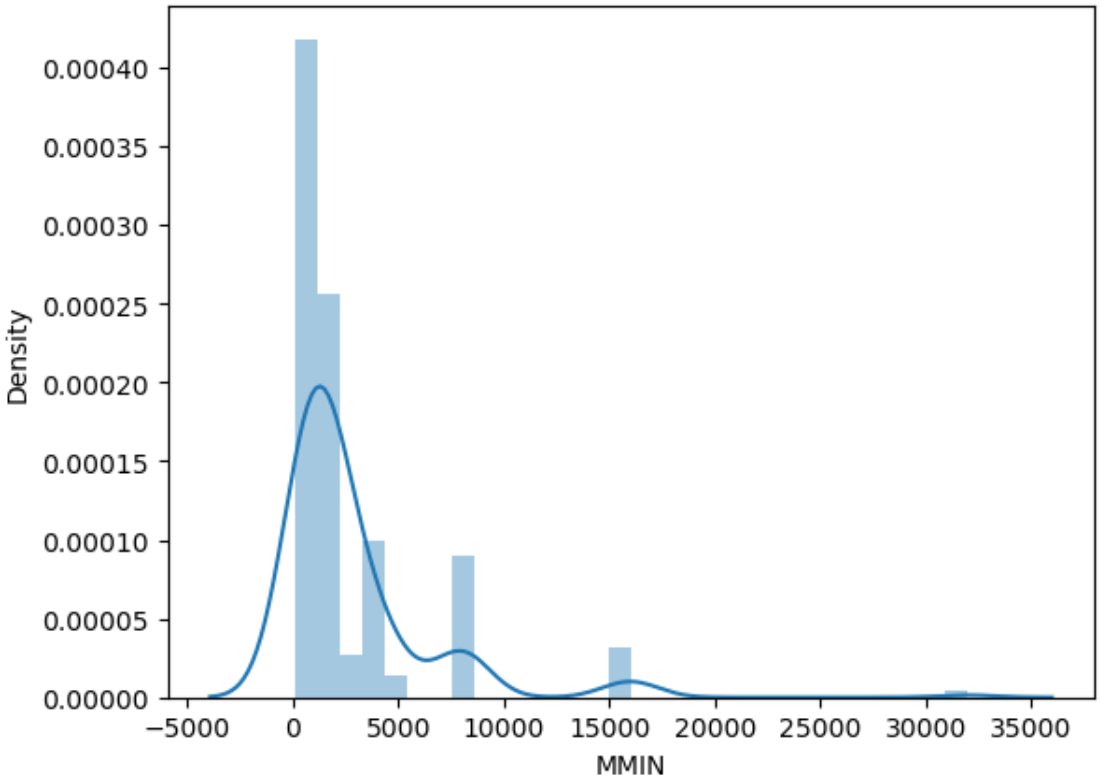
[156]:
```
from scipy.stats import skew
```

[157]:
```
for i in df[colname]:
    print(i)
    print(skew(df[i]))
    sns.distplot(df[i])
    plt.show()
```

```
MYCT
2.5258570096766686
```

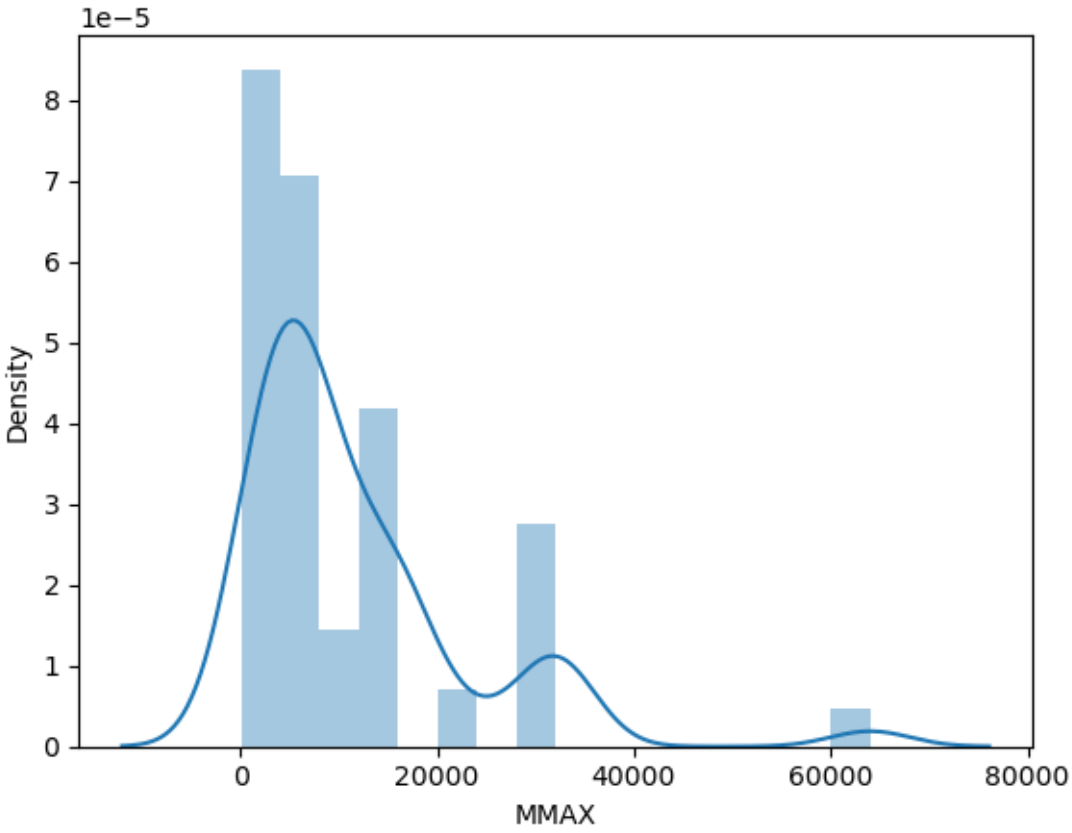



```
MMIN
3.4906489998203925
```

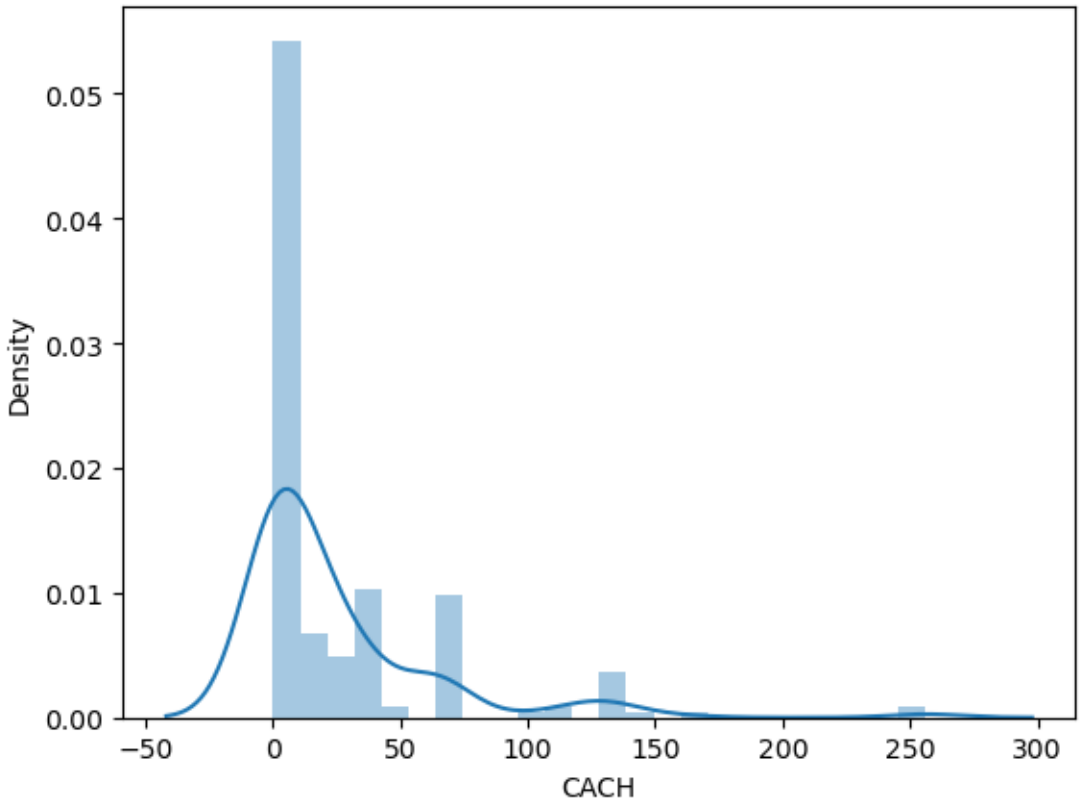


MMAX

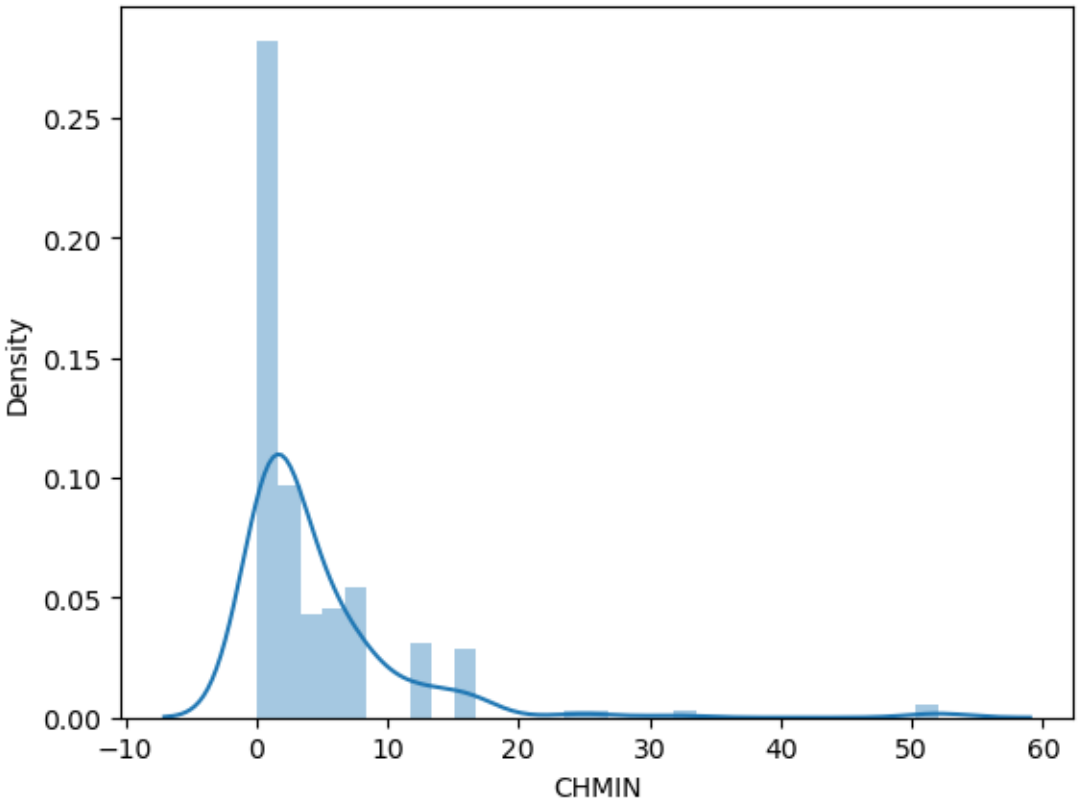
2.1252682972972194



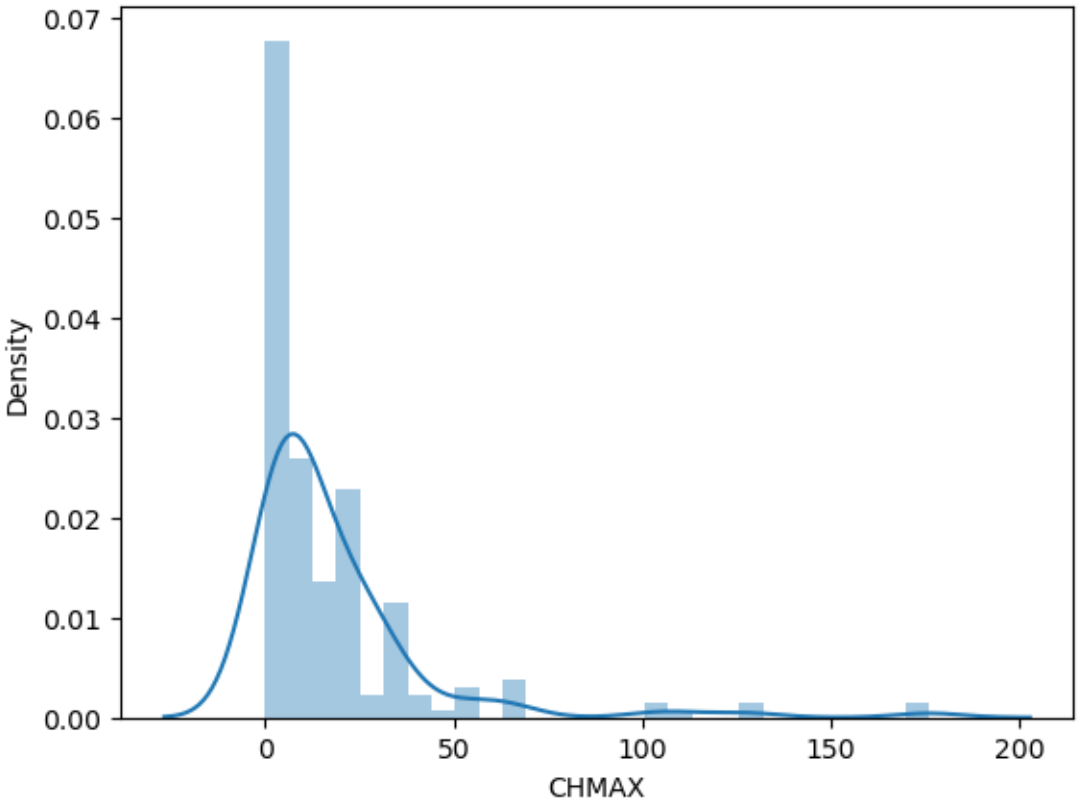

```
CACH
2.8044632567259247
```



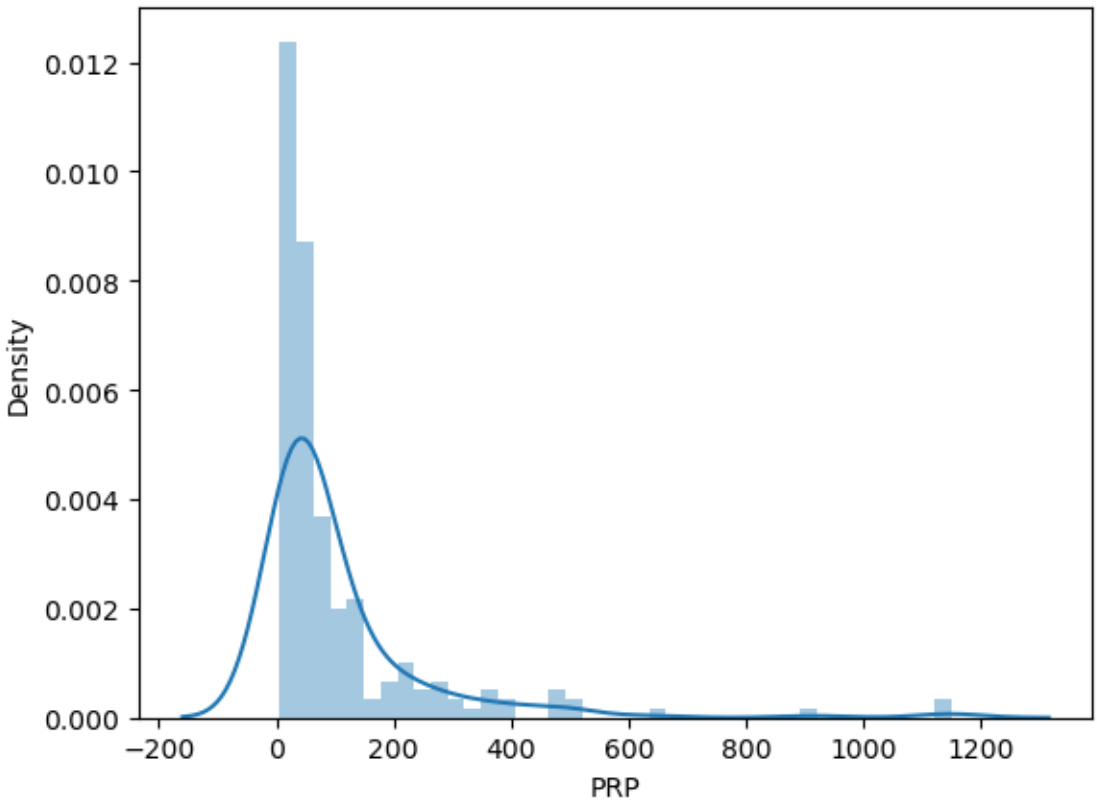

```
CHMIN
3.998370744836009
```



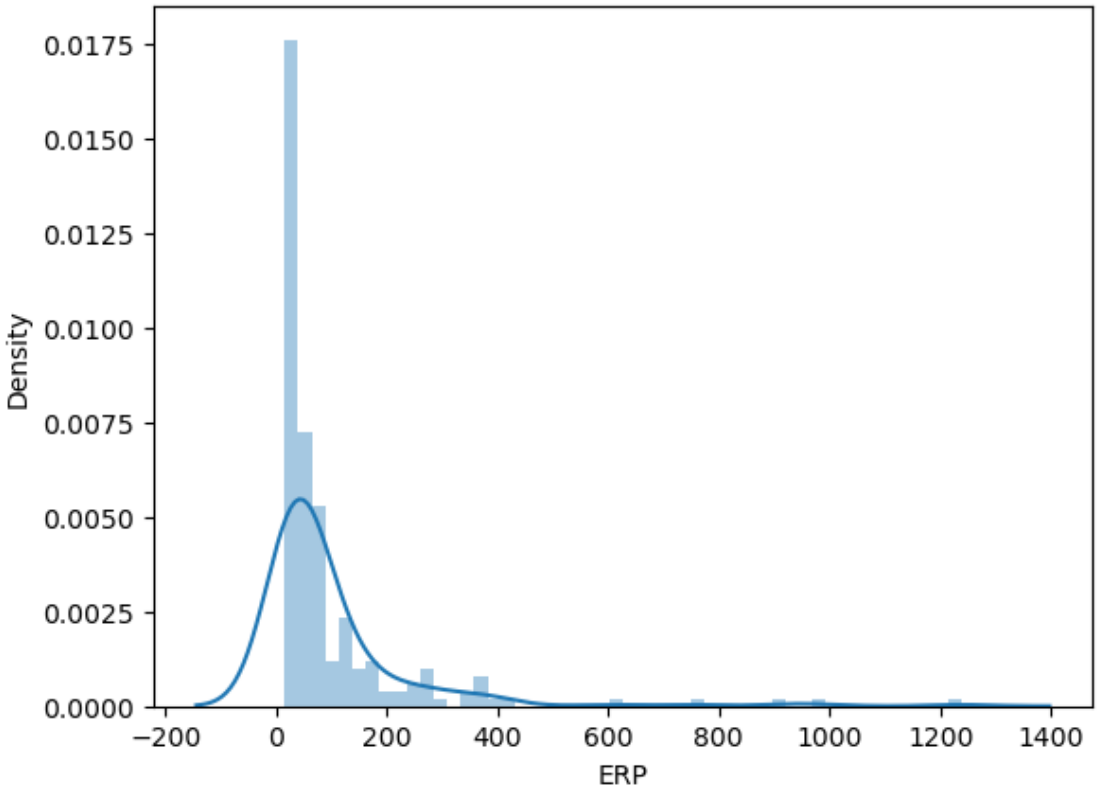

```
CHMAX
3.570045822323458
```




```
PRP
3.864819547035671
```



ERP
4.273072105948175



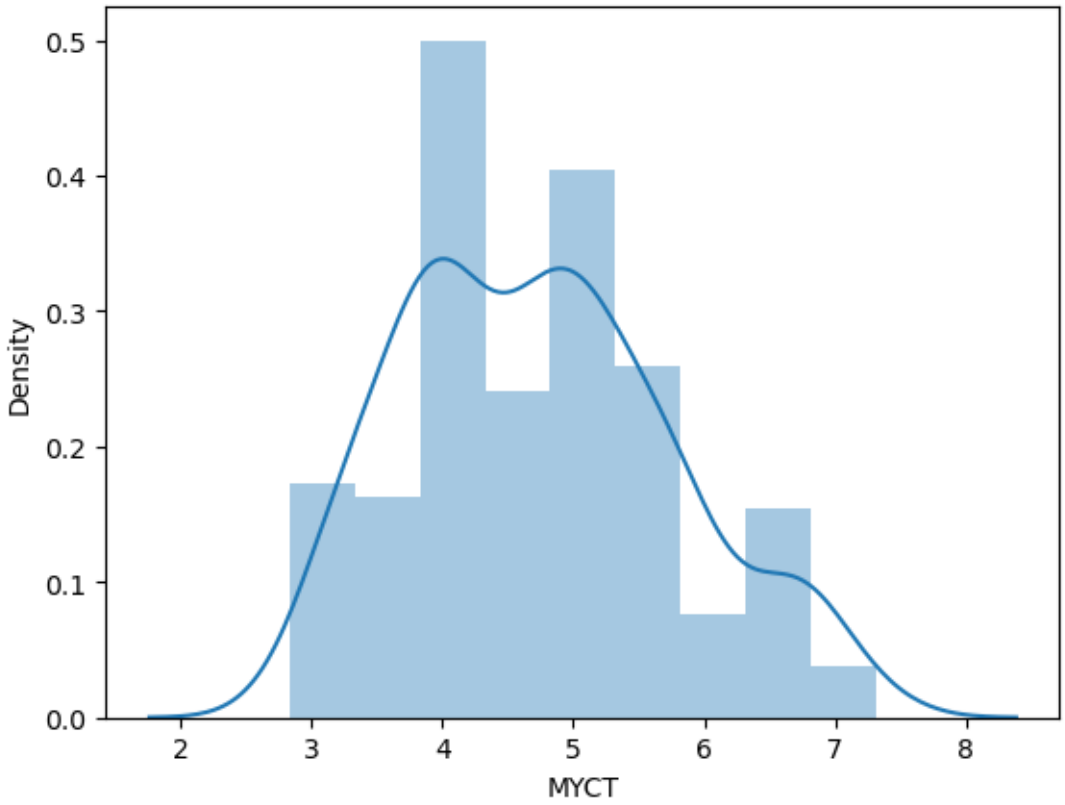
[158]:
```
df.corr().style.background_gradient()
```

[158]:
```
<pandas.io.formats.style.Styler at 0x2428479b910>
```

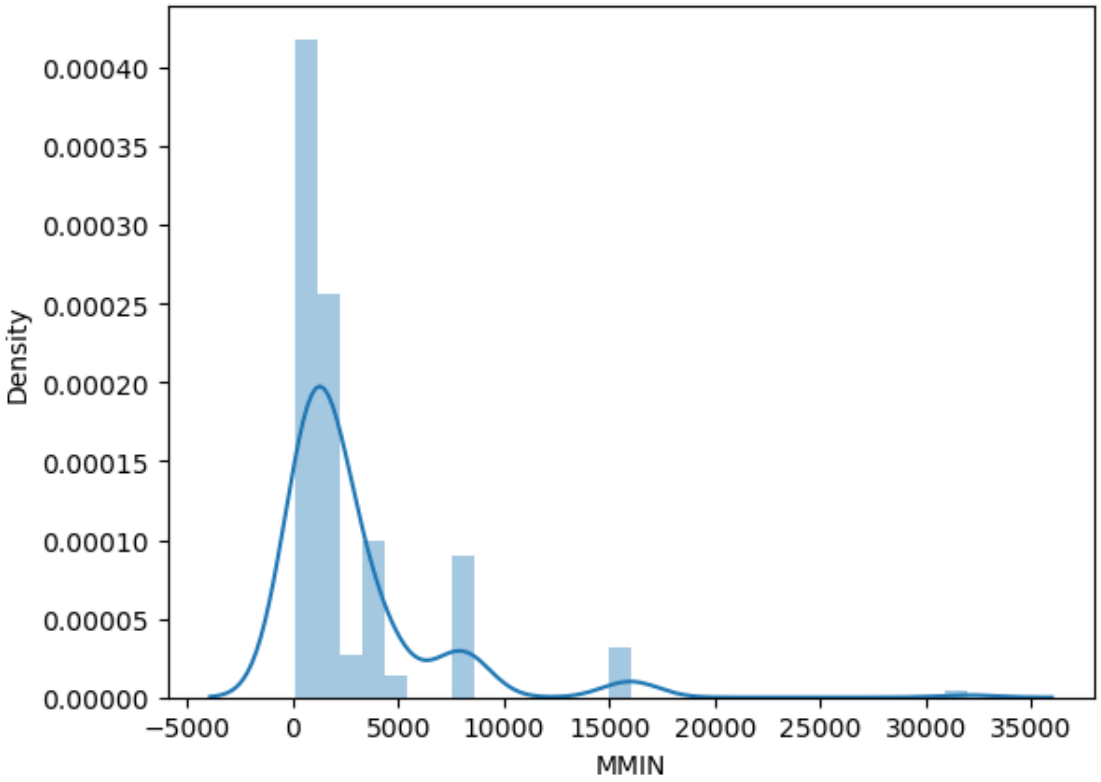
[160]:
```
df[['MYCT','CACH','CHMIN','CHMAX']]=np.log(df[['MYCT','CACH','CHMIN','CHMAX']])
```

[168]:
```
for i in df[colname]:
    print(i)
    print(skew(df[i]))
    sns.distplot(df[i])
    plt.show()
```

```
MYCT
0.39497709484613314
```

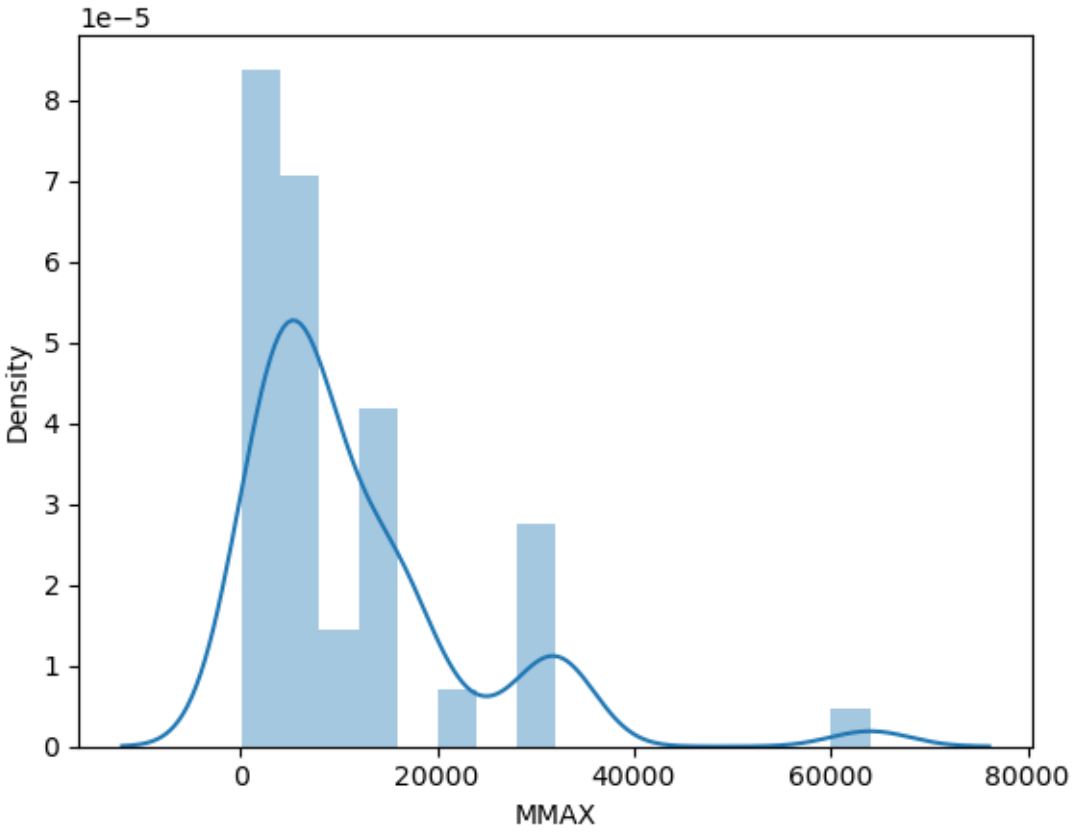



```
MMIN
3.4906489998203925
```

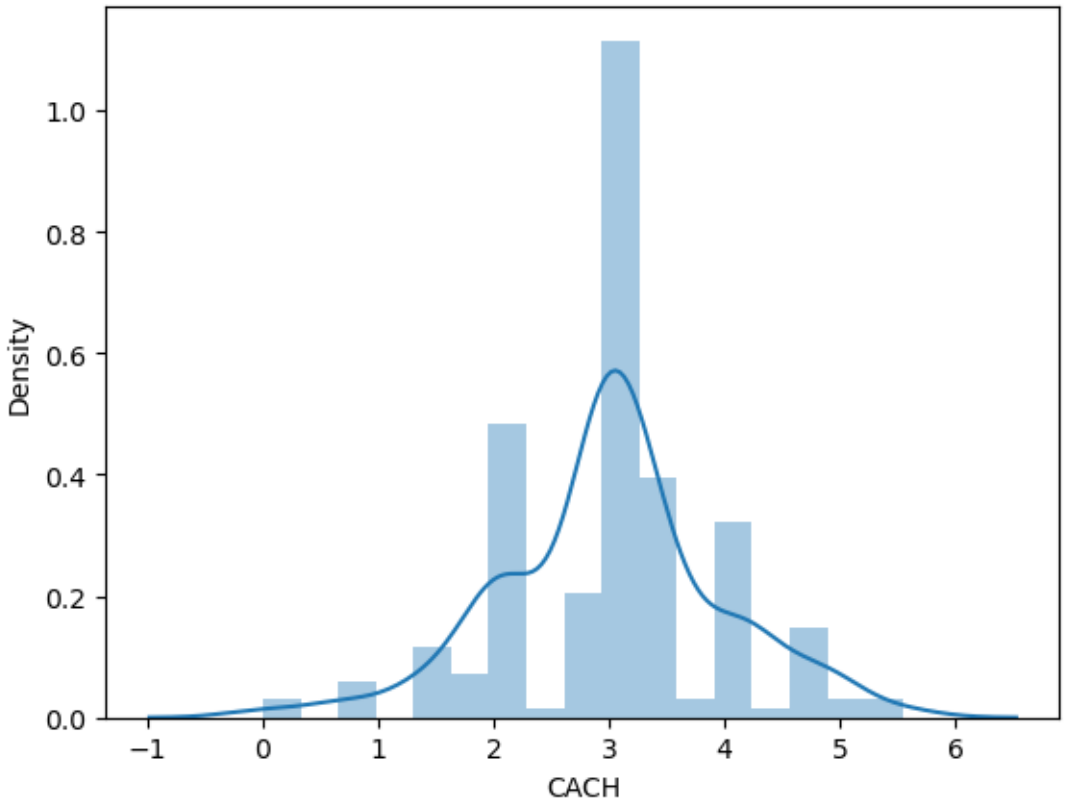


MMAX

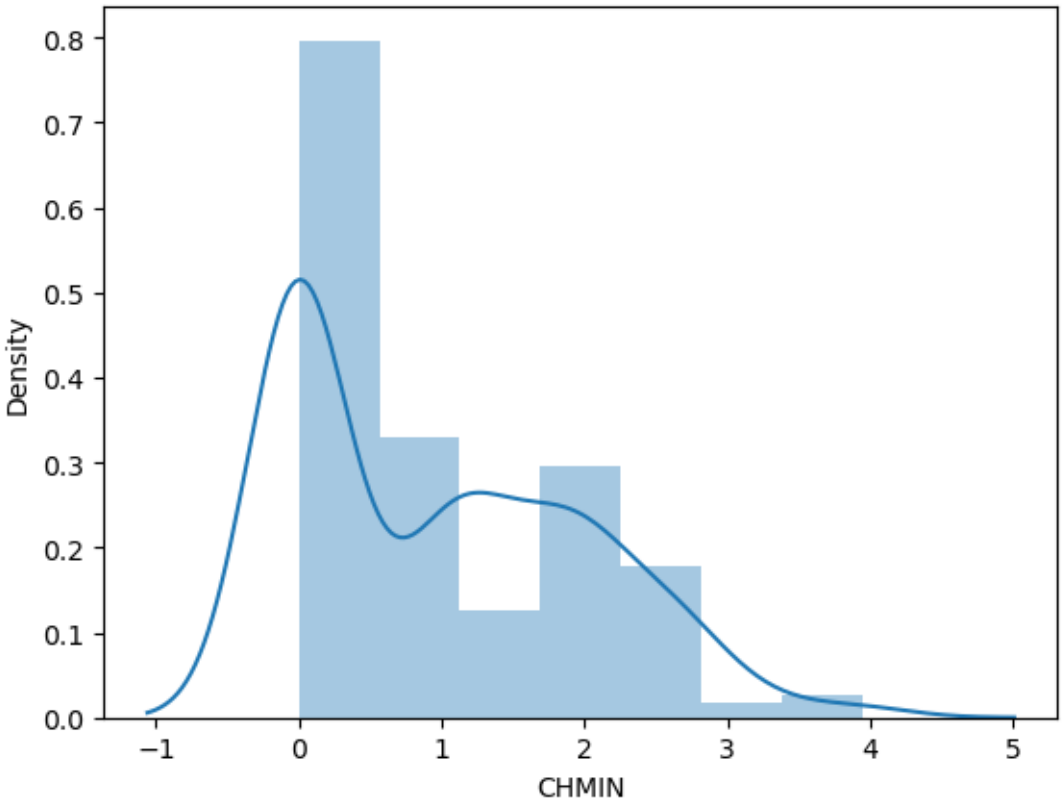
2.1252682972972194



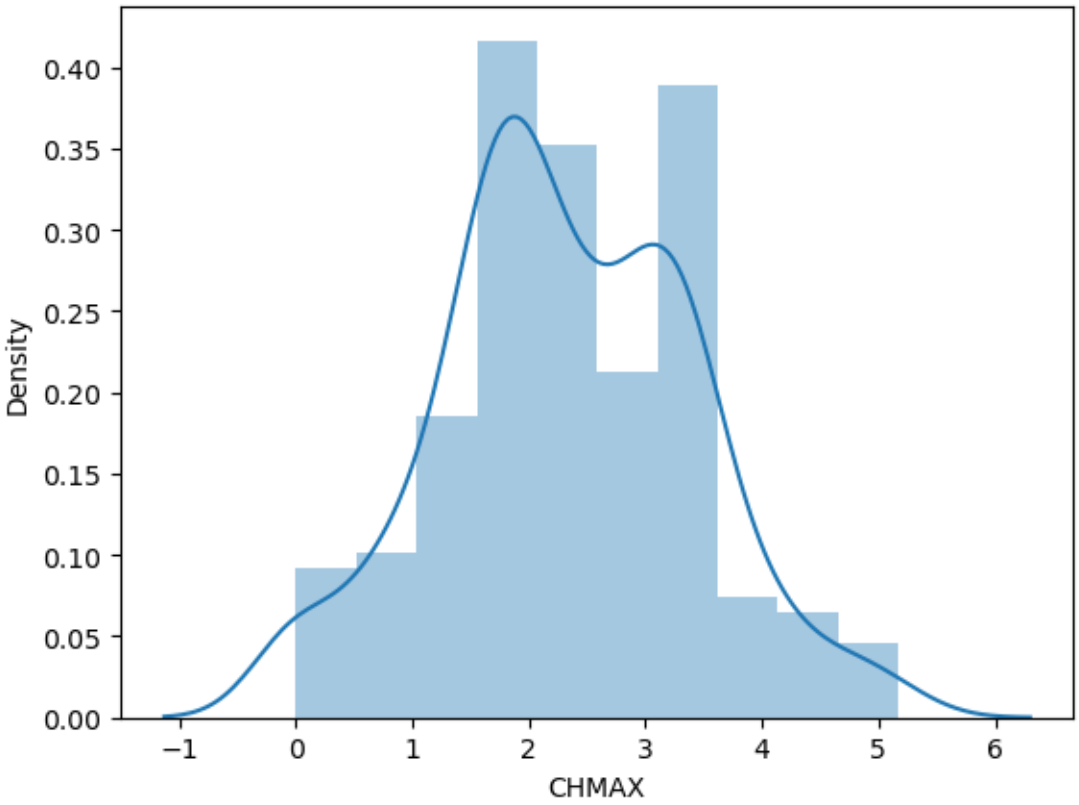

```
CACH
-0.1401785144584389
```



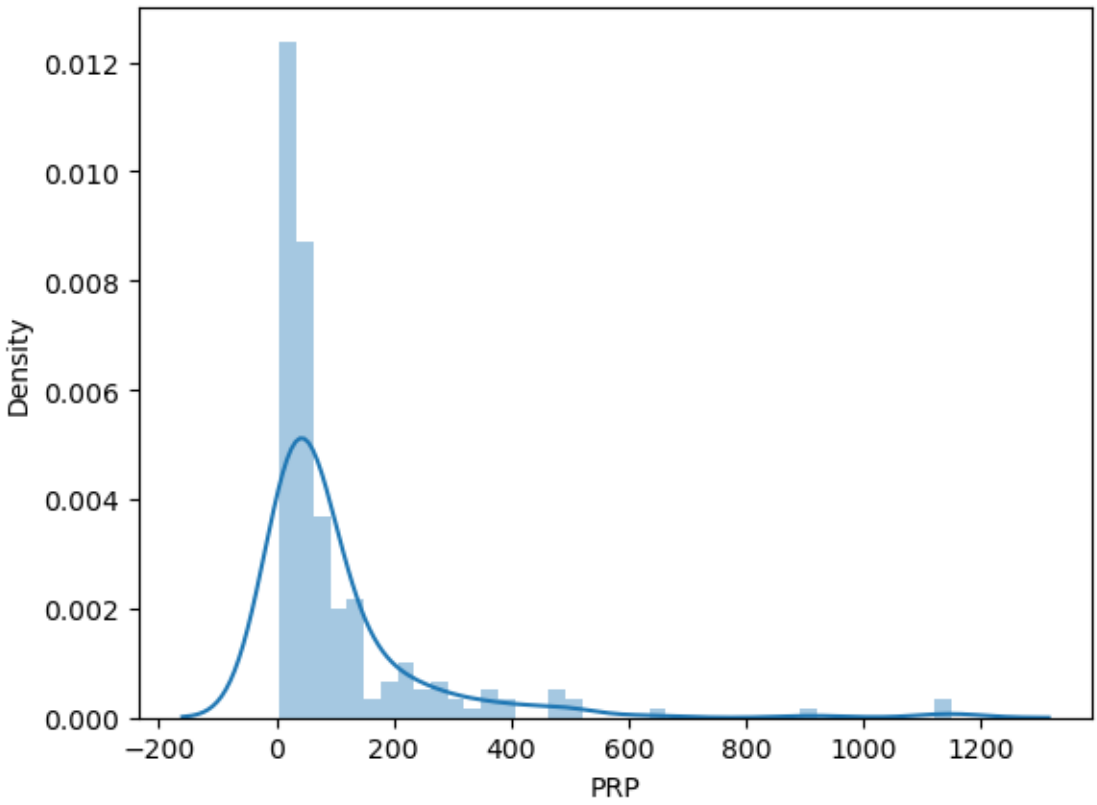

```
CHMIN
0.6142090540769856
```



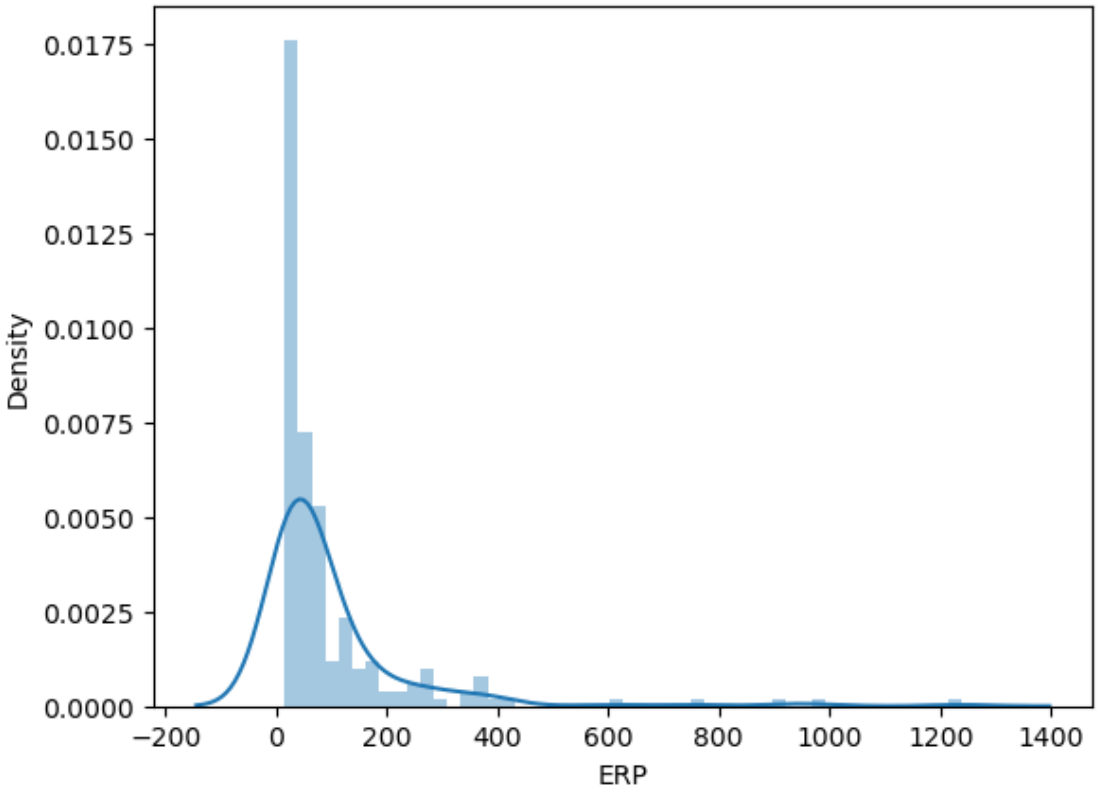

```
CHMAX
0.059470997616673085
```




```
PRP
3.864819547035671
```




```
ERP
4.273072105948175
```



[169]:
```
df.head()
```

[169]:

	Vendors	Model	MYCT	MMIN	MMAX	CACH	CHMIN	CHMAX	PRP	ERP
0	adviser	32/60	4.828314	256	6000	5.545177	2.772589	4.852030	198	199
1	amdahl	470v/7	3.367296	8000	32000	3.465736	2.079442	3.465736	269	253
2	amdahl	470v/7a	3.367296	8000	32000	3.465736	2.079442	3.465736	220	253
3	amdahl	470v/7b	3.367296	8000	32000	3.465736	2.079442	3.465736	172	253
4	amdahl	470v/7c	3.367296	8000	16000	3.465736	2.079442	2.772589	132	132

[170]:
```
catcol=df.select_dtypes(['object']).columns
```

[171]:
```
df[catcol]
```

[171]:

	Vendors	Model
0	adviser	32/60
1	amdahl	470v/7
2	amdahl	470v/7a
3	amdahl	470v/7b
4	amdahl	470v/7c
..	…	…
204	sperry	80/8
205	sperry	90/80-model-3
206	sratus	32
207	wang	vs-100
208	wang	vs-90

[209 rows x 2 columns]

[172]:
```
x=df.iloc[:,:-1]
y=df.iloc[:,-1]
```

[173]:
```
from sklearn.preprocessing import OrdinalEncoder #encoding the categorial value
```

[174]:
```
oe=OrdinalEncoder()
```

[175]:
```
x[catcol]=oe.fit_transform(x[catcol])
```

[176]:
```
x
```

[176]:

	Vendors	Model	MYCT	MMIN	MMAX	CACH	CHMIN	CHMAX	PRP
0	0.0	29.0	4.828314	256	6000	5.545177	2.772589	4.852030	198
1	1.0	62.0	3.367296	8000	32000	3.465736	2.079442	3.465736	269
2	1.0	63.0	3.367296	8000	32000	3.465736	2.079442	3.465736	220
3	1.0	64.0	3.367296	8000	32000	3.465736	2.079442	3.465736	172
4	1.0	65.0	3.367296	8000	16000	3.465736	2.079442	2.772589	132
..	…	…	…	…	…	…	…	…	…
204	27.0	100.0	4.820282	1000	8000	3.012837	0.000000	2.079442	42
205	27.0	109.0	4.584967	1000	8000	3.465736	0.693147	2.079442	46
206	28.0	28.0	4.828314	2000	8000	3.012837	0.693147	2.639057	52
207	29.0	207.0	6.173786	512	8000	3.465736	0.971494	2.317956	67
208	29.0	208.0	6.173786	1000	4000	3.012837	0.971494	2.317956	45

[209 rows x 9 columns]

[177]:
```
y
```

[177]:
```
0      199
1      253
2      253
3      253
4      132
```

```
      …
204     37
205     50
206     41
207     47
208     25
Name: ERP, Length: 209, dtype: int64
```

2 training the model

[178]:
```
from sklearn.model_selection import train_test_split
```

[179]:
```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.3,random_state=1)
```

[180]:
```
xtrain
```

[180]:

	Vendors	Model	MYCT	MMIN	MMAX	CACH	CHMIN	CHMAX	PRP
56	10.0	169.0	5.393628	1000	8000	2.772589	0.000000	0.693147	71
179	25.0	70.0	5.075174	1000	8000	2.772589	0.000000	2.639057	60
55	10.0	168.0	4.700480	1000	12000	2.772589	0.000000	0.693147	60
84	15.0	154.0	5.799093	1000	4000	3.012837	1.098612	1.791759	22
53	10.0	166.0	5.298317	1000	8000	3.012837	0.000000	0.693147	36
..	…	…	…	…	…	…	…	…	…
203	27.0	99.0	5.192957	512	4000	3.012837	0.000000	1.098612	21
137	20.0	185.0	5.010635	512	4000	3.012837	2.079442	4.852030	30
72	16.0	21.0	5.164786	256	2000	3.012837	1.098612	3.178054	22
140	21.0	112.0	4.521789	2000	8000	3.465736	0.000000	1.791759	62
37	8.0	183.0	3.912023	1000	4000	2.079442	0.000000	1.609438	29

[146 rows x 9 columns]

[181]:
```
ytrain
```

[181]:
```
56     42
179    43
55     56
84     25
53     36
       ..
203    24
137    33
72     20
140    53
37     29
Name: ERP, Length: 146, dtype: int64
```

[182]:
```
from sklearn.linear_model import LinearRegression
```

[183]:
```
linreg=LinearRegression()
```

[184]:
```
linreg.fit(xtrain,ytrain)
```

[184]:
```
LinearRegression()
```

[185]:
```
ypred=linreg.predict(xtest)
```

[186]:
```
ypred
```

[186]:
```
array([ 71.09881595,  89.76422957,  25.41421396,  18.15197405,
       211.85664161, 187.33709124,  59.8855257 , 150.28876567,
       199.72471908, 150.22897457,  12.52301752, 139.67596735,
        40.37717789,   7.83430363,  89.25287228, 220.45400742,
        14.56873135,  13.07623384,  50.77784574, 140.56686816,
        36.86734133, 227.31498795,  60.77895385,  14.87916165,
        15.83328941,  41.02384623,  24.10778208, 113.41823494,
        20.79010846, 211.96629983,  14.62318372,  15.87276436,
         8.86369055,  56.86932004,  -2.47867215, 469.47775874,
       207.47294864,  32.4151343 ,  16.22207988, 372.94130696,
       372.12033212,  52.95061996,  36.60098724,  86.52828754,
       129.46335762, 288.8646765 ,  91.41897074,  22.13620622,
        30.99243953,  -4.46405754,  20.24559101,  11.12082273,
        11.81853063,  59.47151299, -12.79083495,  56.71994703,
         1.45008137,  94.76459068,  50.48813859,  53.43457939,
        -5.03319843, 343.15805823,  39.46865836])
```

3 evaluating the model

[187]:
```
from sklearn.metrics import r2_score
```

[216]:
```
r2=r2_score(ytest,ypred)
print(f'accuracy:{r2}')
```

```
accuracy:0.8022673023063933
```

[217]:
```
train=linreg.score(xtrain,ytrain)
test=linreg.score(xtest,ytest)
print(train)
print(test)
```

```
0.9704973330876837
0.8022673023063933
```

[190]:
```
from sklearn.linear_model import Ridge,Lasso
```

[215]:
```
l2=Ridge(alpha=55)
l2.fit(xtrain,ytrain)
```

```
train=l2.score(xtrain,ytrain)
test=l2.score(xtest,ytest)
print(train)
print(test)
```

```
0.9700338483945639
0.801244795002553
```

[211]:
```
l1=Lasso(alpha=1100)
l1.fit(xtrain,ytrain)
train=l1.score(xtrain,ytrain)
test=l1.score(xtest,ytest)
print(train)
print(test)
```

```
0.9609286292482392
0.8031233558099401
```