In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
ds=pd.read_csv("C:/Users/USER/Desktop/Datasets/wbcd.csv")
ds
```

Out[2]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_r
0	87139402	B	12.32	12.39	78.85	464.1	0.1
1	8910251	B	10.60	18.95	69.28	346.4	0.0
2	905520	B	11.04	16.83	70.92	373.2	0.1
3	868871	B	11.28	13.39	73.00	384.8	0.1
4	9012568	B	15.19	13.21	97.65	711.8	0.0
...	...	...	...	...	...	...	
564	911320502	B	13.17	18.22	84.28	537.3	0.0
565	898677	B	10.26	14.71	66.20	321.6	0.0
566	873885	M	15.28	22.41	98.92	710.6	0.0
567	911201	B	14.53	13.98	93.86	644.2	0.1
568	9012795	M	21.37	15.10	141.30	1386.0	0.1

569 rows × 32 columns

In [3]:

```
del ds["id"]
```

In [4]:
```
ds
```

Out[4]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	comp
0	B	12.32	12.39	78.85	464.1	0.10280	
1	B	10.60	18.95	69.28	346.4	0.09688	
2	B	11.04	16.83	70.92	373.2	0.10770	
3	B	11.28	13.39	73.00	384.8	0.11640	
4	B	15.19	13.21	97.65	711.8	0.07963	
...	...	...	...	...	...	...	
564	B	13.17	18.22	84.28	537.3	0.07466	
565	B	10.26	14.71	66.20	321.6	0.09882	
566	M	15.28	22.41	98.92	710.6	0.09057	
567	B	14.53	13.98	93.86	644.2	0.10990	
568	M	21.37	15.10	141.30	1386.0	0.10010	

569 rows × 31 columns

In [5]:
```
ds.describe()
```

Out[5]:

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.0
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.1
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.0
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.0
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.0
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.0
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.1
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.3

8 rows × 30 columns

In [6]:
```
ds.median(numeric_only=True)
```

Out[6]:
```
radius_mean            13.370000
texture_mean           18.840000
perimeter_mean         86.240000
area_mean             551.100000
smoothness_mean         0.095870
compactness_mean        0.092630
concavity_mean          0.061540
points_mean             0.033500
symmetry_mean           0.179200
dimension_mean          0.061540
radius_se               0.324200
texture_se              1.108000
perimeter_se            2.287000
area_se                24.530000
smoothness_se           0.006380
compactness_se          0.020450
concavity_se            0.025890
points_se               0.010930
symmetry_se             0.018730
dimension_se            0.003187
radius_worst           14.970000
texture_worst          25.410000
perimeter_worst        97.660000
area_worst            686.500000
smoothness_worst        0.131300
compactness_worst       0.211900
concavity_worst         0.226700
points_worst            0.099930
symmetry_worst          0.282200
dimension_worst         0.080040
dtype: float64
```

In [7]:

```
ds.dtypes
```

Out[7]:

```
diagnosis             object
radius_mean          float64
texture_mean         float64
perimeter_mean       float64
area_mean            float64
smoothness_mean      float64
compactness_mean     float64
concavity_mean       float64
points_mean          float64
symmetry_mean        float64
dimension_mean       float64
radius_se            float64
texture_se           float64
perimeter_se         float64
area_se              float64
smoothness_se        float64
compactness_se       float64
concavity_se         float64
points_se            float64
symmetry_se          float64
dimension_se         float64
radius_worst         float64
texture_worst        float64
perimeter_worst      float64
area_worst           float64
smoothness_worst     float64
compactness_worst    float64
concavity_worst      float64
points_worst         float64
symmetry_worst       float64
dimension_worst      float64
dtype: object
```

In [8]:

```
ds.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   diagnosis          569 non-null    object 
 1   radius_mean        569 non-null    float64
 2   texture_mean       569 non-null    float64
 3   perimeter_mean     569 non-null    float64
 4   area_mean          569 non-null    float64
 5   smoothness_mean    569 non-null    float64
 6   compactness_mean   569 non-null    float64
 7   concavity_mean     569 non-null    float64
 8   points_mean        569 non-null    float64
 9   symmetry_mean      569 non-null    float64
 10  dimension_mean     569 non-null    float64
 11  radius_se          569 non-null    float64
 12  texture_se         569 non-null    float64
 13  perimeter_se       569 non-null    float64
 14  area_se            569 non-null    float64
 15  smoothness_se      569 non-null    float64
 16  compactness_se     569 non-null    float64
 17  concavity_se       569 non-null    float64
 18  points_se          569 non-null    float64
 19  symmetry_se        569 non-null    float64
 20  dimension_se       569 non-null    float64
 21  radius_worst       569 non-null    float64
 22  texture_worst      569 non-null    float64
 23  perimeter_worst    569 non-null    float64
 24  area_worst         569 non-null    float64
 25  smoothness_worst   569 non-null    float64
 26  compactness_worst  569 non-null    float64
 27  concavity_worst    569 non-null    float64
 28  points_worst       569 non-null    float64
 29  symmetry_worst     569 non-null    float64
 30  dimension_worst    569 non-null    float64
dtypes: float64(30), object(1)
memory usage: 137.9+ KB
```

In [9]:

```
ds.isnull().sum()
```

Out[9]:

```
diagnosis            0
radius_mean          0
texture_mean         0
perimeter_mean       0
area_mean            0
smoothness_mean      0
compactness_mean     0
concavity_mean       0
points_mean          0
symmetry_mean        0
dimension_mean       0
radius_se            0
texture_se           0
perimeter_se         0
area_se              0
smoothness_se        0
compactness_se       0
concavity_se         0
points_se            0
symmetry_se          0
dimension_se         0
radius_worst         0
texture_worst        0
perimeter_worst      0
area_worst           0
smoothness_worst     0
compactness_worst    0
concavity_worst      0
points_worst         0
symmetry_worst       0
dimension_worst      0
dtype: int64
```

In [10]:

```
ds["diagnosis"].value_counts()
```

Out[10]:

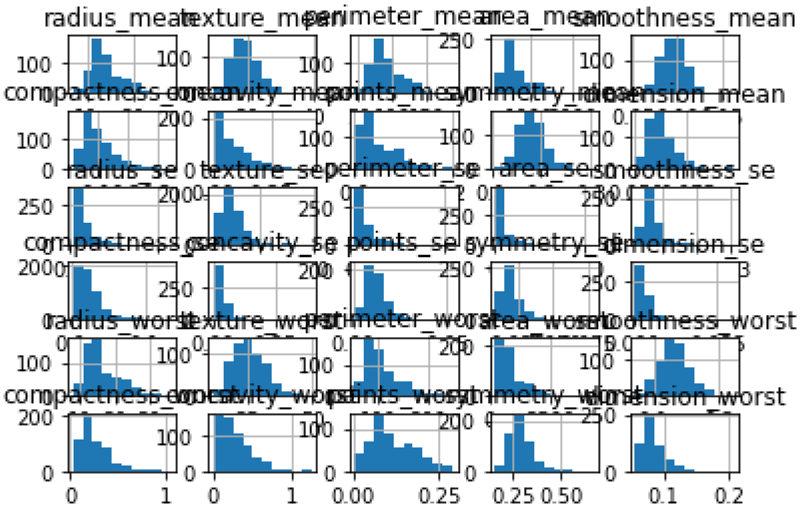
```
B    357
M    212
Name: diagnosis, dtype: int64
```

In [15]:

```
ds.hist()
```

Out[15]:

```
array([[<AxesSubplot:title={'center':'radius_mean'}>,
        <AxesSubplot:title={'center':'texture_mean'}>,
        <AxesSubplot:title={'center':'perimeter_mean'}>,
        <AxesSubplot:title={'center':'area_mean'}>,
        <AxesSubplot:title={'center':'smoothness_mean'}>],
       [<AxesSubplot:title={'center':'compactness_mean'}>,
        <AxesSubplot:title={'center':'concavity_mean'}>,
        <AxesSubplot:title={'center':'points_mean'}>,
        <AxesSubplot:title={'center':'symmetry_mean'}>,
        <AxesSubplot:title={'center':'dimension_mean'}>],
       [<AxesSubplot:title={'center':'radius_se'}>,
        <AxesSubplot:title={'center':'texture_se'}>,
        <AxesSubplot:title={'center':'perimeter_se'}>,
        <AxesSubplot:title={'center':'area_se'}>,
        <AxesSubplot:title={'center':'smoothness_se'}>],
       [<AxesSubplot:title={'center':'compactness_se'}>,
        <AxesSubplot:title={'center':'concavity_se'}>,
        <AxesSubplot:title={'center':'points_se'}>,
        <AxesSubplot:title={'center':'symmetry_se'}>,
        <AxesSubplot:title={'center':'dimension_se'}>],
       [<AxesSubplot:title={'center':'radius_worst'}>,
        <AxesSubplot:title={'center':'texture_worst'}>,
        <AxesSubplot:title={'center':'perimeter_worst'}>,
        <AxesSubplot:title={'center':'area_worst'}>,
        <AxesSubplot:title={'center':'smoothness_worst'}>],
       [<AxesSubplot:title={'center':'compactness_worst'}>,
        <AxesSubplot:title={'center':'concavity_worst'}>,
        <AxesSubplot:title={'center':'points_worst'}>,
        <AxesSubplot:title={'center':'symmetry_worst'}>,
        <AxesSubplot:title={'center':'dimension_worst'}>]], dtype=object)
```



In [16]:

```
x=pd.DataFrame(ds.iloc[:,1:])
y=pd.DataFrame(ds.iloc[:,0])
```

In [17]:

```
x
```

Out[17]:

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_me
0	12.32	12.39	78.85	464.1	0.10280	0.069
1	10.60	18.95	69.28	346.4	0.09688	0.114
2	11.04	16.83	70.92	373.2	0.10770	0.078
3	11.28	13.39	73.00	384.8	0.11640	0.113
4	15.19	13.21	97.65	711.8	0.07963	0.069
...	...	...	...	...	...	
564	13.17	18.22	84.28	537.3	0.07466	0.059
565	10.26	14.71	66.20	321.6	0.09882	0.091
566	15.28	22.41	98.92	710.6	0.09057	0.105
567	14.53	13.98	93.86	644.2	0.10990	0.092
568	21.37	15.10	141.30	1386.0	0.10010	0.151

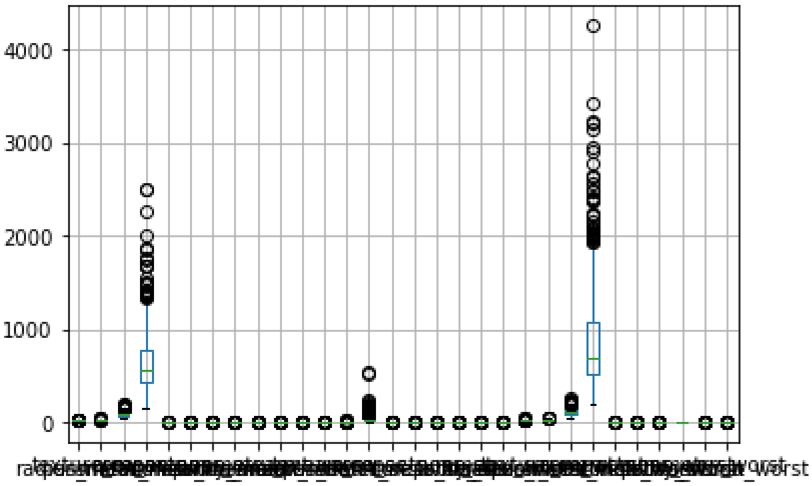
569 rows × 30 columns

In [18]:

```
boxplot=x.boxplot()
boxplot
```

Out[18]:

```
<AxesSubplot:>
```



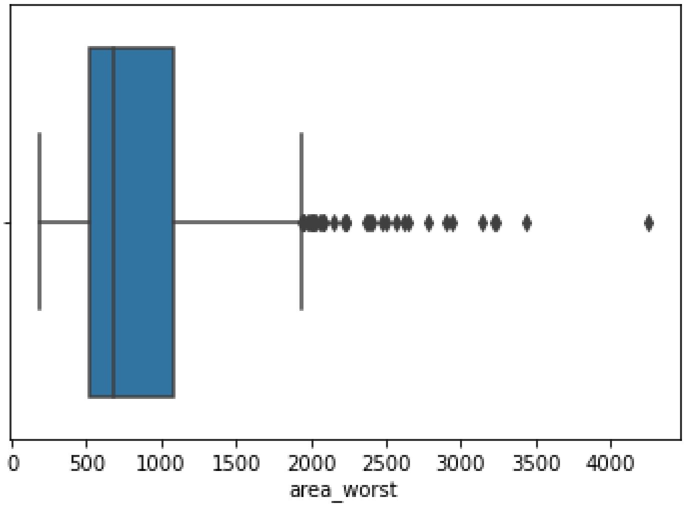
In [19]:

```
sns.boxplot(x["area_worst"])
```

```
C:\Users\USER\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarn
ing: Pass the following variable as a keyword arg: x. From version 0.12, the on
ly valid positional argument will be `data`, and passing other arguments withou
t an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[19]:

```
<AxesSubplot:xlabel='area_worst'>
```



In [20]:

```
from feature_engine.outliers import Winsorizer
win=Winsorizer(capping_method="iqr", tail="both", fold=1.5, variables=["area_wors
new_area_worst=win.fit_transform(x[["area_worst"]])
```

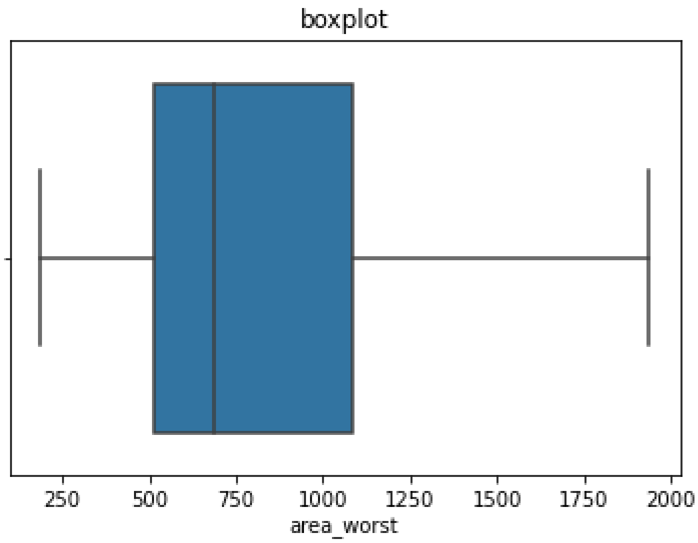
In [21]:

```
sns.boxplot(new_area_worst.area_worst)
plt.title("boxplot")
plt.show
```

```
C:\Users\USER\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarn
ing: Pass the following variable as a keyword arg: x. From version 0.12, the on
ly valid positional argument will be `data`, and passing other arguments withou
t an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[21]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [22]:

```
x.insert(loc=23,column="new_area_worst", value=new_area_worst)
```

In [23]:

```
del x["area_worst"]
```

In [24]:
```
x
```

Out[24]:

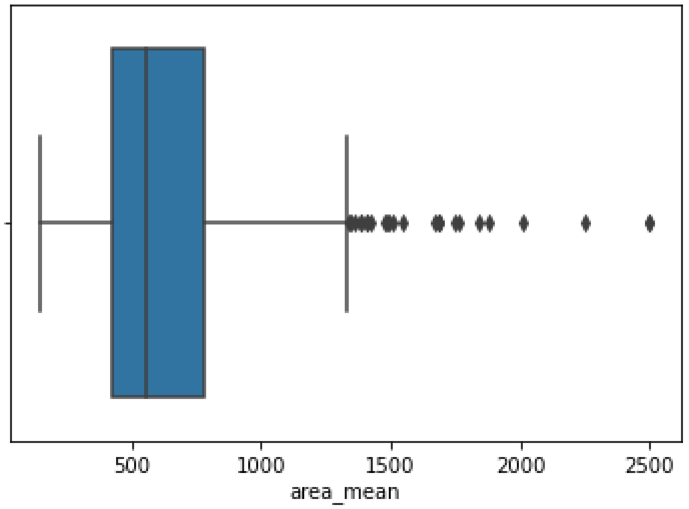
	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_me
0	12.32	12.39	78.85	464.1	0.10280	0.069
1	10.60	18.95	69.28	346.4	0.09688	0.114
2	11.04	16.83	70.92	373.2	0.10770	0.078
3	11.28	13.39	73.00	384.8	0.11640	0.113
4	15.19	13.21	97.65	711.8	0.07963	0.069
...	...	...	...	...	...	
564	13.17	18.22	84.28	537.3	0.07466	0.059
565	10.26	14.71	66.20	321.6	0.09882	0.091
566	15.28	22.41	98.92	710.6	0.09057	0.105
567	14.53	13.98	93.86	644.2	0.10990	0.092
568	21.37	15.10	141.30	1386.0	0.10010	0.151

569 rows × 30 columns

In [25]:
```
sns.boxplot(x["area_mean"])
```

```
C:\Users\USER\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarn
ing: Pass the following variable as a keyword arg: x. From version 0.12, the on
ly valid positional argument will be `data`, and passing other arguments withou
t an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[25]:
```
<AxesSubplot:xlabel='area_mean'>
```

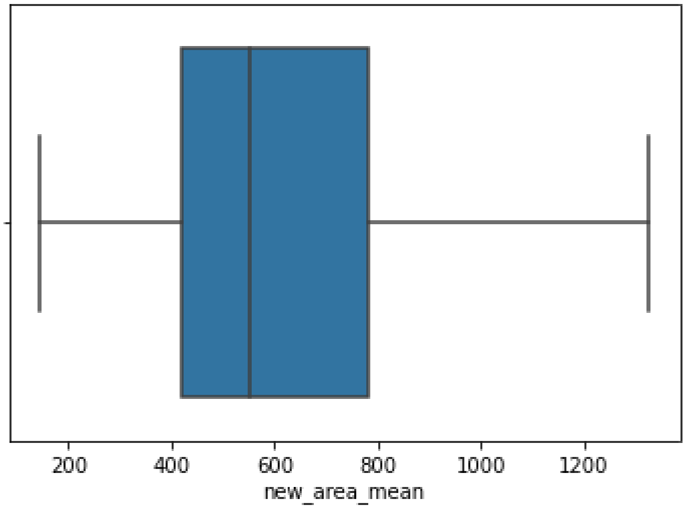


In [28]:
```
from feature_engine.outliers import Winsorizer
win=Winsorizer(capping_method="iqr", tail="both", fold=1.5, variables=["area_mear
new_area_mean=win.fit_transform(x[["area_mean"]])
x.insert(loc=3,column="new_area_mean", value=new_area_mean)
```

In [29]:
```
sns.boxplot(x["new_area_mean"])
```

```
C:\Users\USER\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarn
ing: Pass the following variable as a keyword arg: x. From version 0.12, the on
ly valid positional argument will be `data`, and passing other arguments withou
t an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[29]:
```
<AxesSubplot:xlabel='new_area_mean'>
```



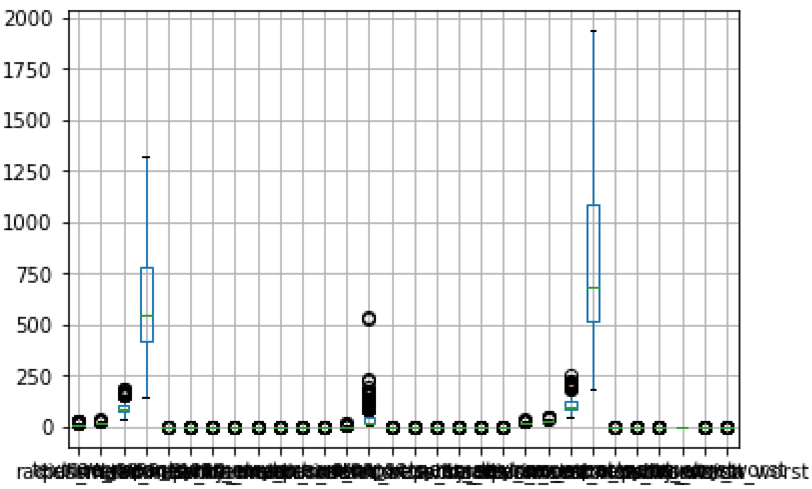
In [30]:
```
del x["area_mean"]
x
```

Out[30]:

	radius_mean	texture_mean	perimeter_mean	new_area_mean	smoothness_mean	compact
0	12.32	12.39	78.85	464.1	0.10280	
1	10.60	18.95	69.28	346.4	0.09688	
2	11.04	16.83	70.92	373.2	0.10770	
3	11.28	13.39	73.00	384.8	0.11640	
4	15.19	13.21	97.65	711.8	0.07963	
...	...	...	...	...	...	
564	13.17	18.22	84.28	537.3	0.07466	
565	10.26	14.71	66.20	321.6	0.09882	
566	15.28	22.41	98.92	710.6	0.09057	
567	14.53	13.98	93.86	644.2	0.10990	
568	21.37	15.10	141.30	1326.3	0.10010	

In [31]:
```
boxplot=x.boxplot()
boxplot
```

Out[31]:
```
<AxesSubplot:>
```



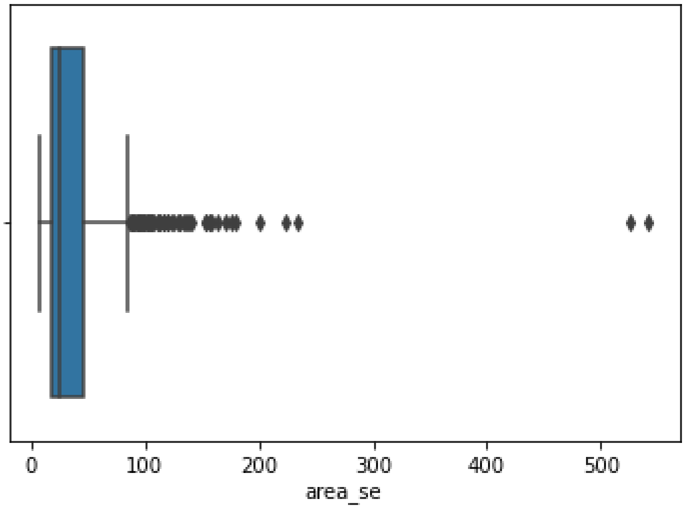
In [35]:
```
names=x.columns
names
```

Out[35]:
```
Index(['radius_mean', 'texture_mean', 'perimeter_mean', 'new_area_mean',
       'smoothness_mean', 'compactness_mean', 'concavity_mean', 'points_mean',
       'symmetry_mean', 'dimension_mean', 'radius_se', 'texture_se',
       'perimeter_se', 'area_se', 'smoothness_se', 'compactness_se',
       'concavity_se', 'points_se', 'symmetry_se', 'dimension_se',
       'radius_worst', 'texture_worst', 'perimeter_worst', 'new_area_worst',
       'smoothness_worst', 'compactness_worst', 'concavity_worst',
       'points_worst', 'symmetry_worst', 'dimension_worst'],
      dtype='object')
```

In [38]:
```
sns.boxplot(x["area_se"])
```

```
C:\Users\USER\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarn
ing: Pass the following variable as a keyword arg: x. From version 0.12, the on
ly valid positional argument will be `data`, and passing other arguments withou
t an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[38]:
```
<AxesSubplot:xlabel='area_se'>
```



In [40]:
```
from feature_engine.outliers import Winsorizer
win=Winsorizer(capping_method="iqr", tail="both", fold=1.5, variables=["area_se"]
new_area_sc=win.fit_transform(x[["area_se"]])
x.insert(loc=13,column="new_area_se", value=new_area_sc)
```

In [42]:

```
del x["area_se"]
x
```

Out[42]:

	radius_mean	texture_mean	perimeter_mean	new_area_mean	smoothness_mean	compactnes
0	12.32	12.39	78.85	464.1	0.10280	
1	10.60	18.95	69.28	346.4	0.09688	
2	11.04	16.83	70.92	373.2	0.10770	
3	11.28	13.39	73.00	384.8	0.11640	
4	15.19	13.21	97.65	711.8	0.07963	
...	...	...	...	...	...	
564	13.17	18.22	84.28	537.3	0.07466	
565	10.26	14.71	66.20	321.6	0.09882	
566	15.28	22.41	98.92	710.6	0.09057	
567	14.53	13.98	93.86	644.2	0.10990	
568	21.37	15.10	141.30	1326.3	0.10010	

569 rows × 30 columns

In [43]:

```
names=x.columns
names
```

Out[43]:

```
Index(['radius_mean', 'texture_mean', 'perimeter_mean', 'new_area_mean',
       'smoothness_mean', 'compactness_mean', 'concavity_mean', 'points_mean',
       'symmetry_mean', 'dimension_mean', 'radius_se', 'texture_se',
       'perimeter_se', 'new_area_se', 'smoothness_se', 'compactness_se',
       'concavity_se', 'points_se', 'symmetry_se', 'dimension_se',
       'radius_worst', 'texture_worst', 'perimeter_worst', 'new_area_worst',
       'smoothness_worst', 'compactness_worst', 'concavity_worst',
       'points_worst', 'symmetry_worst', 'dimension_worst'],
      dtype='object')
```

In [44]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test=train_test_split(x,y, train_size=0.80)
```

In [46]:

```
from sklearn.preprocessing import StandardScaler
scale = StandardScaler()
x_train = scale.fit_transform(x_train)
x_test = scale.transform(x_test)
```

In [47]:

```
from sklearn.neighbors import KNeighborsClassifier
clf=KNeighborsClassifier(n_neighbors=3)
```

In [49]:

```
clf.fit(x_train, y_train)
```

```
C:\Users\USER\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:
179: DataConversionWarning: A column-vector y was passed when a 1d array was ex
pected. Please change the shape of y to (n_samples,), for example using ravel
().
  return self._fit(X, y)
```

Out[49]:

```
KNeighborsClassifier(n_neighbors=3)
```

In [50]:

```
y_pred=clf.predict(x_test)
y_pred
```

Out[50]:

```
array(['M', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'B',
       'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'M', 'B', 'M', 'B',
       'B', 'M', 'M', 'B', 'M', 'B', 'M', 'B', 'B', 'B', 'B', 'M', 'B',
       'M', 'B', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'B', 'M', 'M', 'M',
       'B', 'M', 'M', 'M', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B',
       'B', 'M', 'M', 'B', 'B', 'M', 'M', 'B', 'M', 'M', 'M', 'B', 'B',
       'M', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'M', 'M',
       'B', 'B', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M', 'B',
       'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B'], dtype=object)
```

In [51]:

```
from sklearn.metrics import confusion_matrix, accuracy_score
print(confusion_matrix(y_test, y_pred))
```

```
[[75  0]
 [ 4 35]]
```

In [53]:

```
accuracy_score(y_test, y_pred)
```

Out[53]:

```
0.9649122807017544
```

In [ ]:

```

```