Outliner

Detect

- Maximum & Minimum values
- Percentile
- Mean & standard deviation
- Inter Qualities Range

Eliminate

- Trimming
- Capping

1. Using maximum and minimum values

In [1]:
```
# Import data
import pandas as pd
df = pd.read_excel('dataset.xlsx', sheet_name='outliner')
```

In [2]:
```
df
```

Out[2]:

	ID	salary
0	1001	21652
1	1002	20007
2	1003	29464
3	1004	25998
4	1005	21565
5	1006	57801
6	1007	60100
7	1008	29361
8	1009	27654
9	1010	23086
10	1011	26780
11	1012	21144
12	1013	21986
13	1014	23036
14	1015	29674
15	1016	29365
16	1017	25259
17	1018	26575
18	1019	25366
19	1020	22169
20	1021	26183
21	1022	23010
22	1023	25931
23	1024	25474
24	1025	29748
25	1026	25092
26	1027	28403
27	1028	21464
28	1029	4780
29	1030	20167

In [3]:
```
df.describe()
```

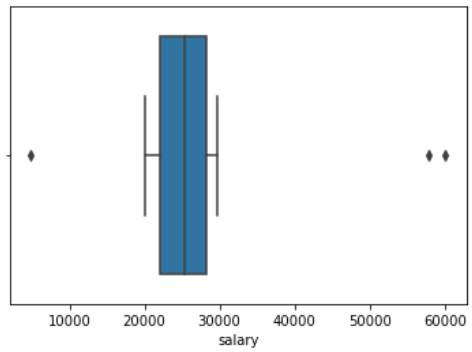
Out[3]:

	ID	salary
count	30.000000	30.000000
mean	1015.500000	26609.800000
std	8.803408	9994.181705
min	1001.000000	4780.000000
25%	1008.250000	22031.750000
50%	1015.500000	25420.000000
75%	1022.750000	28215.750000
max	1030.000000	60100.000000

In [4]:
```
import seaborn as sns
sns.boxplot(df['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

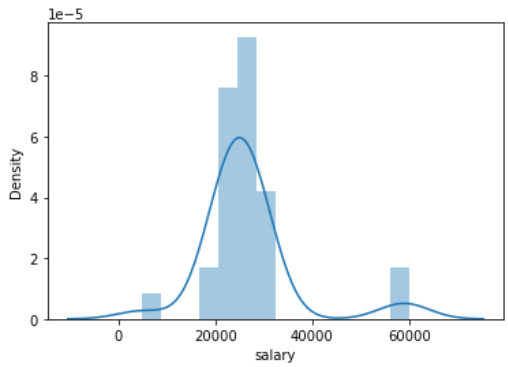
Out[4]:
```
<AxesSubplot:xlabel='salary'>
```



In [5]:
```
sns.distplot(df['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[5]:
```
<AxesSubplot:xlabel='salary', ylabel='Density'>
```



Find boundary for outliner

In [6]:
```
import numpy as np
lower_limit = np.sort(df['salary'].values)[1]
upper_limit = np.sort(df['salary'].values)[-2]
```

In [7]:
```
# 0 1 2 3 ... 199 200 201 202
```

In [8]:
```
lower_limit
```

Out[8]:
```
20007
```

In [9]:
```
upper_limit
```

Out[9]:
```
57801
```

Trimming the outliner values

In [10]:
```
df_trim = df.loc[df['salary'] >= lower_limit]
```

In [11]:
```
df.describe()
```

Out[11]:

	ID	salary
count	30.000000	30.000000
mean	1015.500000	26609.800000
std	8.803408	9994.181705
min	1001.000000	4780.000000
25%	1008.250000	22031.750000
50%	1015.500000	25420.000000
75%	1022.750000	28215.750000
max	1030.000000	60100.000000

In [12]:
```
df_trim.describe()
```

Out[12]:

	ID	salary
count	29.000000	29.000000
mean	1015.034483	27362.551724
std	8.575225	9265.245705
min	1001.000000	20007.000000
25%	1008.000000	22169.000000
50%	1015.000000	25474.000000
75%	1022.000000	28403.000000
max	1030.000000	60100.000000

In [13]:
```
df_trim = df_trim.loc[df_trim['salary'] <= upper_limit]
```

In [14]:
```
df_trim.describe()
```

Out[14]:

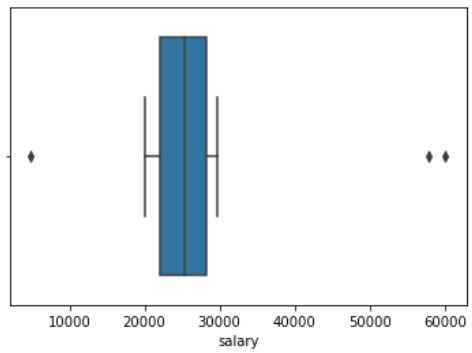
	ID	salary
count	28.000000	28.000000
mean	1015.321429	26193.357143
std	8.589630	6921.883752
min	1001.000000	20007.000000
25%	1008.750000	22123.250000
50%	1015.500000	25420.000000
75%	1022.250000	27841.250000
max	1030.000000	57801.000000

In [15]:
```
# Option : use only one command
# df_trim = df[(df['salary'] <= upper_limit) & (df['salary'] >= lower_limit)]
```

In [16]:
```
sns.boxplot(df['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

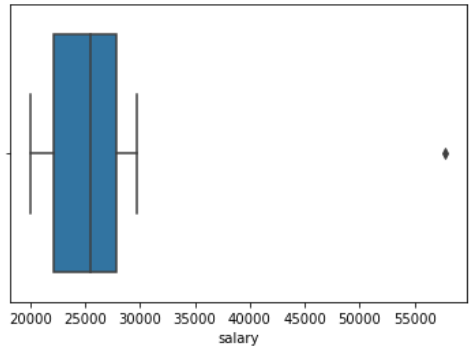
Out[16]:
```
<AxesSubplot:xlabel='salary'>
```



In [17]:
```
sns.boxplot(df_trim['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

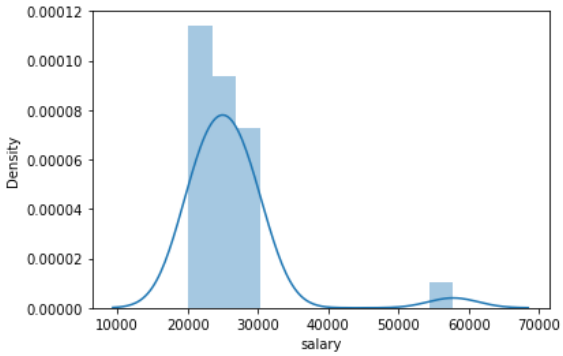
Out[17]:
```
<AxesSubplot:xlabel='salary'>
```



In [18]:
```
sns.distplot(df_trim['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[18]:
```
<AxesSubplot:xlabel='salary', ylabel='Density'>
```



Clipping the outliner values

In [19]:
```
df_clip = df.copy()
```

In [20]:
```
df_clip['salary'] = df['salary'].replace( df['salary'].min() , lower_limit)
```

In [21]:
```
df_clip.describe()
```

Out[21]:

	ID	salary
count	30.000000	30.000000
mean	1015.500000	27117.366667
std	8.803408	9202.612922
min	1001.000000	20007.000000
25%	1008.250000	22031.750000
50%	1015.500000	25420.000000
75%	1022.750000	28215.750000
max	1030.000000	60100.000000

In [22]:
```
df_clip['salary'] = df_clip['salary'].replace( df_clip['salary'].max() , upper_limit)
```

In [23]:
```
df_clip.describe()
```

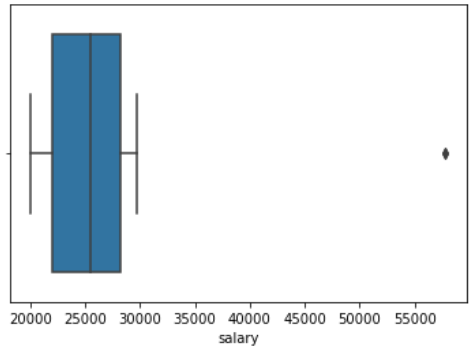
Out[23]:

	ID	salary
count	30.000000	30.000000
mean	1015.500000	27040.733333
std	8.803408	8923.833887
min	1001.000000	20007.000000
25%	1008.250000	22031.750000
50%	1015.500000	25420.000000
75%	1022.750000	28215.750000
max	1030.000000	57801.000000

In [24]:
```
sns.boxplot(df_clip['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

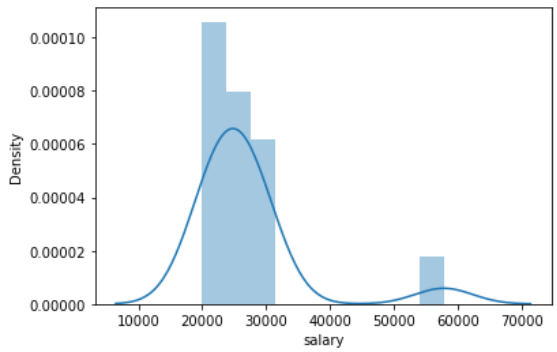
Out[24]:
```
<AxesSubplot:xlabel='salary'>
```



In [25]:
```
sns.distplot(df_clip['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[25]:
```
<AxesSubplot:xlabel='salary', ylabel='Density'>
```



2. Using percentile

In [26]:
```
import pandas as pd
df = pd.read_excel('dataset.xlsx', sheet_name='outliner')
```

In [27]:
```
df
```

Out[27]:

	ID	salary
0	1001	21652
1	1002	20007
2	1003	29464
3	1004	25998
4	1005	21565
5	1006	57801
6	1007	60100
7	1008	29361
8	1009	27654
9	1010	23086
10	1011	26780
11	1012	21144
12	1013	21986
13	1014	23036
14	1015	29674
15	1016	29365
16	1017	25259
17	1018	26575
18	1019	25366
19	1020	22169
20	1021	26183
21	1022	23010
22	1023	25931
23	1024	25474
24	1025	29748
25	1026	25092
26	1027	28403
27	1028	21464
28	1029	4780
29	1030	20167

In [28]:
```
df.describe()
```

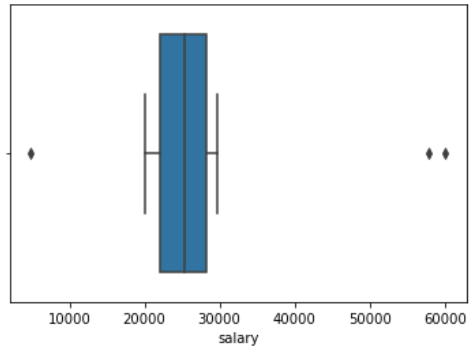
Out[28]:

	ID	salary
count	30.000000	30.000000
mean	1015.500000	26609.800000
std	8.803408	9994.181705
min	1001.000000	4780.000000
25%	1008.250000	22031.750000
50%	1015.500000	25420.000000
75%	1022.750000	28215.750000
max	1030.000000	60100.000000

In [29]:
```
import seaborn as sns
sns.boxplot(df['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

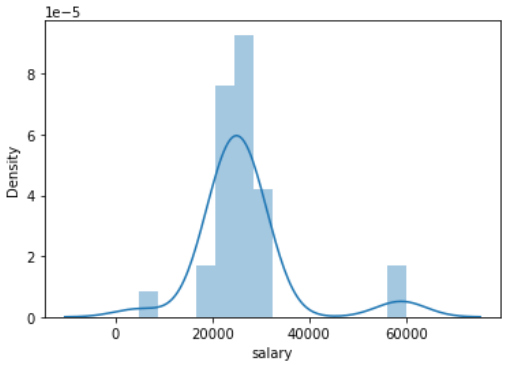
Out[29]:
```
<AxesSubplot:xlabel='salary'>
```



In [30]:
```
sns.distplot(df['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[30]:
```
<AxesSubplot:xlabel='salary', ylabel='Density'>
```



Find boundary for outliner

In [31]:
```
upper_limit = df['salary'].quantile(0.99)
lower_limit = df['salary'].quantile(0.01)
```

In [32]:
```
upper_limit
```

Out[32]:
```
59433.29
```

In [33]:
```
lower_limit
```

Out[33]:
```
9195.83
```

Trimming the outliner values

In [34]:
```
df_trim = df[(df['salary'] <= upper_limit) & (df['salary'] >= lower_limit)]
```

In [35]:
```
df_trim.describe()
```

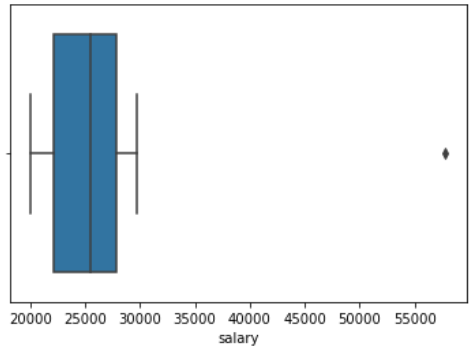
Out[35]:

	ID	salary
count	28.000000	28.000000
mean	1015.321429	26193.357143
std	8.589630	6921.883752
min	1001.000000	20007.000000
25%	1008.750000	22123.250000
50%	1015.500000	25420.000000
75%	1022.250000	27841.250000
max	1030.000000	57801.000000

In [36]:
```
sns.boxplot(df_trim['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

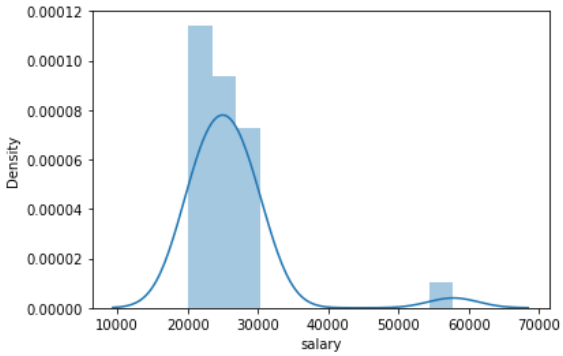
Out[36]:
```
<AxesSubplot:xlabel='salary'>
```



In [37]:
```
sns.distplot(df_trim['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[37]:
```
<AxesSubplot:xlabel='salary', ylabel='Density'>
```



Clipping the outliner values

In [38]:
```
df_clip = df.copy()
```

In [39]:
```
df_clip['salary'] = df['salary'].replace( df['salary'].loc[df['salary'] < lower_limit] , lower_limit)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [39], in <cell line: 1>()
----> 1 df_clip['salary'] = df['salary'].replace( df['salary'].loc[df['salary'] < lower_limit] , lower_limit)

File ~/opt/anaconda3/lib/python3.9/site-packages/pandas/core/series.py:4960, in Series.replace(self, to_replace, value, inplace, limit, regex, method)
   4945 @doc(
   4946     NDFrame.replace,  # type: ignore[has-type]
   4947     klass=_shared_doc_kwargs["klass"],
   (...)
   4958     method: str | lib.NoDefault = lib.no_default,
   4959 ):
-> 4960     return super().replace(
   4961         to_replace=to_replace,
   4962         value=value,
   4963         inplace=inplace,
   4964         limit=limit,
   4965         regex=regex,
   4966         method=method,
   4967     )

File ~/opt/anaconda3/lib/python3.9/site-packages/pandas/core/generic.py:6679, in NDFrame.replace(self, to_replace, value, inplace, limit, regex, method)
   6676 elif not is_list_like(value):
   6677     # Operate column-wise
   6678     if self.ndim == 1:
-> 6679         raise ValueError(
   6680             "Series.replace cannot use dict-like to_replace "
   6681             "and non-None value"
   6682         )
   6683     mapping = {
   6684         col: (to_rep, value) for col, to_rep in to_replace.items()
   6685     }
   6686     return self._replace_columnwise(mapping, inplace, regex)

ValueError: Series.replace cannot use dict-like to_replace and non-None value
```

In [40]:
```
df_clip.describe()
```

Out[40]:

	ID	salary
count	30.000000	30.000000
mean	1015.500000	26609.800000
std	8.803408	9994.181705
min	1001.000000	4780.000000
25%	1008.250000	22031.750000
50%	1015.500000	25420.000000
75%	1022.750000	28215.750000
max	1030.000000	60100.000000

In [41]:
```
df_clip['salary'] = df_clip['salary'].replace( df_clip['salary'].loc[df_clip['salary'] > upper_limit] , upper_limit)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [41], in <cell line: 1>()
----> 1 df_clip['salary'] = df_clip['salary'].replace( df_clip['salary'].loc[df_clip['salary'] > upper_limit] , upper_l
imit)

File ~/opt/anaconda3/lib/python3.9/site-packages/pandas/core/series.py:4960, in Series.replace(self, to_replace, value,
inplace, limit, regex, method)
   4945 @doc(
   4946     NDFrame.replace,  # type: ignore[has-type]
   4947     klass=_shared_doc_kwargs["klass"],
   (...)
   4958     method: str | lib.NoDefault = lib.no_default,
   4959 ):
-> 4960     return super().replace(
   4961         to_replace=to_replace,
   4962         value=value,
   4963         inplace=inplace,
   4964         limit=limit,
   4965         regex=regex,
   4966         method=method,
   4967     )

File ~/opt/anaconda3/lib/python3.9/site-packages/pandas/core/generic.py:6679, in NDFrame.replace(self, to_replace, valu
e, inplace, limit, regex, method)
   6676 elif not is_list_like(value):
   6677     # Operate column-wise
   6678     if self.ndim == 1:
-> 6679         raise ValueError(
   6680             "Series.replace cannot use dict-like to_replace "
   6681             "and non-None value"
   6682         )
   6683     mapping = {
   6684         col: (to_rep, value) for col, to_rep in to_replace.items()
   6685     }
   6686     return self._replace_columnwise(mapping, inplace, regex)

ValueError: Series.replace cannot use dict-like to_replace and non-None value
```

In [42]:
```
df_clip.describe()
```

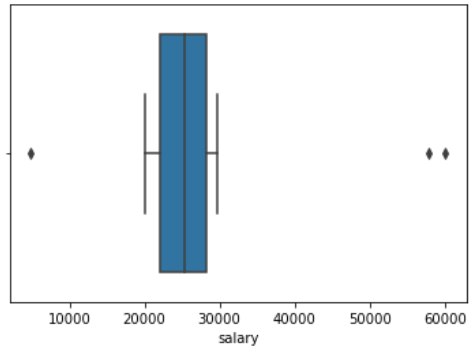
Out[42]:

	ID	salary
count	30.000000	30.000000
mean	1015.500000	26609.800000
std	8.803408	9994.181705
min	1001.000000	4780.000000
25%	1008.250000	22031.750000
50%	1015.500000	25420.000000
75%	1022.750000	28215.750000
max	1030.000000	60100.000000

In [43]:
```
sns.boxplot(df['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

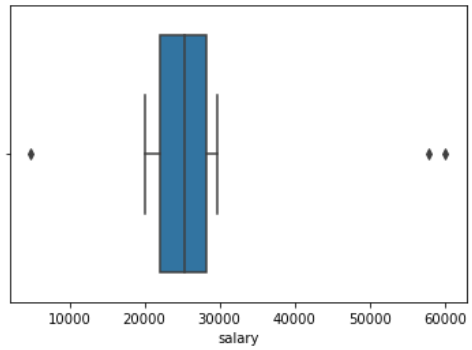
Out[43]:
```
<AxesSubplot:xlabel='salary'>
```



In [44]:
```
sns.boxplot(df_clip['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

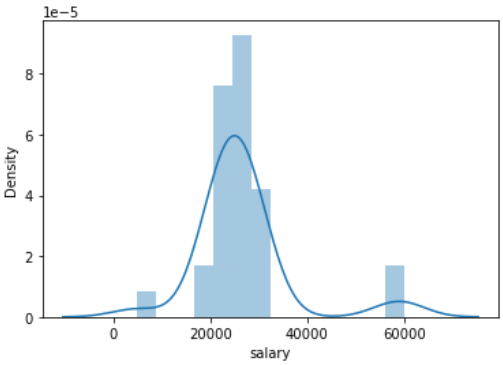
Out[44]:
```
<AxesSubplot:xlabel='salary'>
```



In [45]:
```
sns.distplot(df_clip['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[45]:
```
<AxesSubplot:xlabel='salary', ylabel='Density'>
```



Change the boundary to the 10th percentile

In [46]:
```
upper_limit = df['salary'].quantile(0.90)
lower_limit = df['salary'].quantile(0.10)
```

In [47]:
```
upper_limit
```

Out[47]:
```
29681.4
```

In [48]:
```
lower_limit
```

Out[48]:
```
21046.3
```

In [49]:
```
df_trim = df[(df['salary'] <= upper_limit) & (df['salary'] >= lower_limit)]
```

In [50]:
```
df_trim.describe()
```

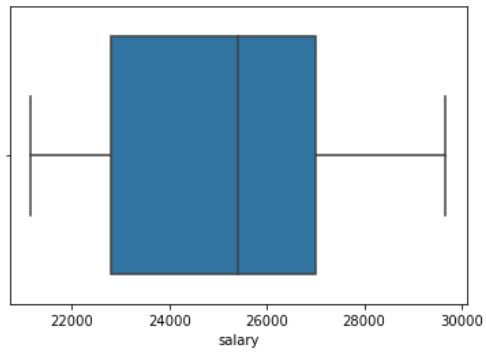
Out[50]:

	ID	salary
count	24.000000	24.00000
mean	1015.250000	25237.12500
std	7.853274	2839.35947
min	1001.000000	21144.00000
25%	1009.750000	22799.75000
50%	1015.500000	25420.00000
75%	1021.250000	26998.50000
max	1028.000000	29674.00000

In [51]:
```
sns.boxplot(df_trim['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

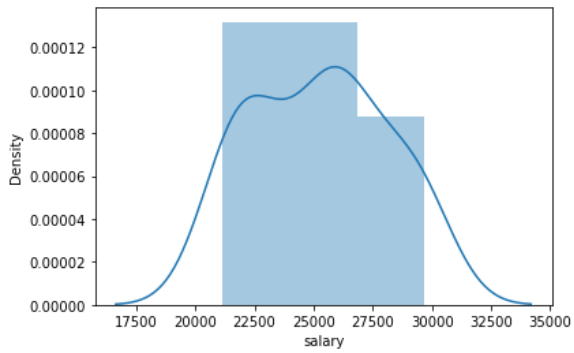
Out[51]:
```
<AxesSubplot:xlabel='salary'>
```



In [52]:
```
sns.distplot(df_trim['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[52]:
```
<AxesSubplot:xlabel='salary', ylabel='Density'>
```



3. Using the Inter-Quantile Range (IQR)

In [53]:
```
import pandas as pd
df = pd.read_excel('dataset.xlsx', sheet_name='outliner')
```

In [54]:
```
df.describe()
```

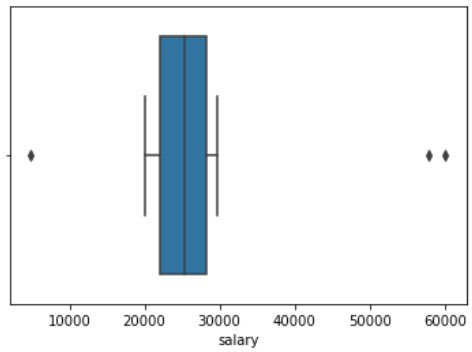
Out[54]:

	ID	salary
count	30.000000	30.000000
mean	1015.500000	26609.800000
std	8.803408	9994.181705
min	1001.000000	4780.000000
25%	1008.250000	22031.750000
50%	1015.500000	25420.000000
75%	1022.750000	28215.750000
max	1030.000000	60100.000000

In [55]:
```
import seaborn as sns
sns.boxplot(df['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

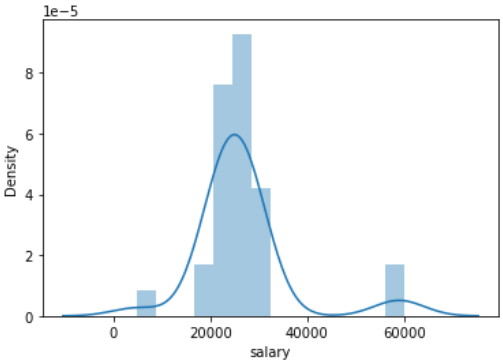
Out[55]:
```
<AxesSubplot:xlabel='salary'>
```



In [56]:
```
sns.distplot(df['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[56]:
```
<AxesSubplot:xlabel='salary', ylabel='Density'>
```



Find boundary for outliner

In [57]:
```
q1 = df['salary'].quantile(0.25)
q3 = df['salary'].quantile(0.75)
iqr = q3-q1
```

In [58]:
```
upper_limit = q3 + ( 1.5 * iqr )
lower_limit = q1 - ( 1.5 * iqr )
```

In [59]:
```
upper_limit
```

Out[59]:
```
37491.75
```

In [60]:
```
lower_limit
```

Out[60]:
```
12755.75
```

Trimming the outliner values

In [61]:
```
df_trim = df[(df['salary'] <= upper_limit) & (df['salary'] >= lower_limit)]
```

In [62]:
```
df_trim.describe()
```

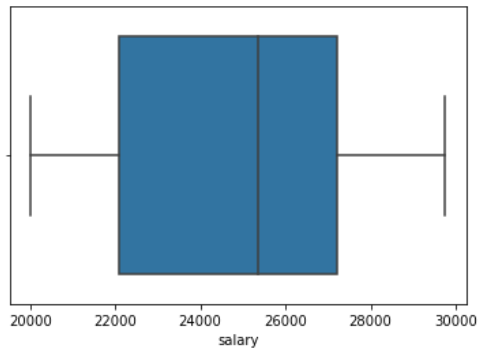
Out[62]:

	ID	salary
count	27.000000	27.000000
mean	1015.666667	25022.703704
std	8.553002	3147.600635
min	1001.000000	20007.000000
25%	1009.500000	22077.500000
50%	1016.000000	25366.000000
75%	1022.500000	27217.000000
max	1030.000000	29748.000000

In [63]:
```
sns.boxplot(df_trim['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

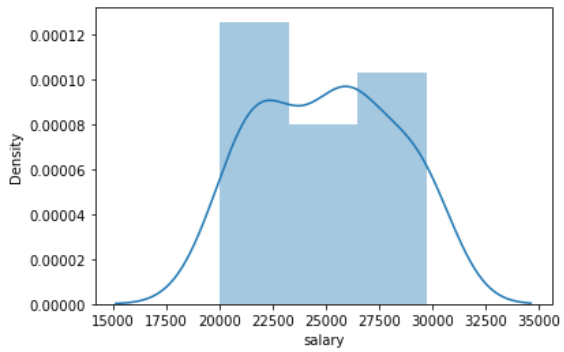
Out[63]:
```
<AxesSubplot:xlabel='salary'>
```



In [64]:
```
sns.distplot(df_trim['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[64]:
```
<AxesSubplot:xlabel='salary', ylabel='Density'>
```



Clipping the outliner values

In [65]:
```
df_clip = df.copy()
```

In [66]:
```
df_clip['salary'] = df['salary'].replace( df['salary'].loc[df['salary'] < lower_limit] , lower_limit)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [66], in <cell line: 1>()
----> 1 df_clip['salary'] = df['salary'].replace( df['salary'].loc[df['salary'] < lower_limit] , lower_limit)

File ~/opt/anaconda3/lib/python3.9/site-packages/pandas/core/series.py:4960, in Series.replace(self, to_replace, value,
inplace, limit, regex, method)
   4945 @doc(
   4946     NDFrame.replace,  # type: ignore[has-type]
   4947     klass=_shared_doc_kwargs["klass"],
   (...)
   4958     method: str | lib.NoDefault = lib.no_default,
   4959 ):
-> 4960     return super().replace(
   4961         to_replace=to_replace,
   4962         value=value,
   4963         inplace=inplace,
   4964         limit=limit,
   4965         regex=regex,
   4966         method=method,
   4967     )

File ~/opt/anaconda3/lib/python3.9/site-packages/pandas/core/generic.py:6679, in NDFrame.replace(self, to_replace, valu
e, inplace, limit, regex, method)
   6676 elif not is_list_like(value):
   6677     # Operate column-wise
   6678     if self.ndim == 1:
-> 6679         raise ValueError(
   6680             "Series.replace cannot use dict-like to_replace "
   6681             "and non-None value"
   6682         )
   6683     mapping = {
   6684         col: (to_rep, value) for col, to_rep in to_replace.items()
   6685     }
   6686     return self._replace_columnwise(mapping, inplace, regex)

ValueError: Series.replace cannot use dict-like to_replace and non-None value
```

In [67]:
```
df_clip.describe()
```

Out[67]:

	ID	salary
count	30.000000	30.000000
mean	1015.500000	26609.800000
std	8.803408	9994.181705
min	1001.000000	4780.000000
25%	1008.250000	22031.750000
50%	1015.500000	25420.000000
75%	1022.750000	28215.750000
max	1030.000000	60100.000000

In [68]:
```
df_clip['salary'] = df_clip['salary'].replace( df_clip['salary'].loc[df_clip['salary'] > upper_limit] , upper_limit)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [68], in <cell line: 1>()
----> 1 df_clip['salary'] = df_clip['salary'].replace( df_clip['salary'].loc[df_clip['salary'] > upper_limit] , upper_l
imit)

File ~/opt/anaconda3/lib/python3.9/site-packages/pandas/core/series.py:4960, in Series.replace(self, to_replace, value,
inplace, limit, regex, method)
   4945 @doc(
   4946     NDFrame.replace,  # type: ignore[has-type]
   4947     klass=_shared_doc_kwargs["klass"],
   (...)
   4958     method: str | lib.NoDefault = lib.no_default,
   4959 ):
-> 4960     return super().replace(
   4961         to_replace=to_replace,
   4962         value=value,
   4963         inplace=inplace,
   4964         limit=limit,
   4965         regex=regex,
   4966         method=method,
   4967     )

File ~/opt/anaconda3/lib/python3.9/site-packages/pandas/core/generic.py:6679, in NDFrame.replace(self, to_replace, valu
e, inplace, limit, regex, method)
   6676 elif not is_list_like(value):
   6677     # Operate column-wise
   6678     if self.ndim == 1:
-> 6679         raise ValueError(
   6680             "Series.replace cannot use dict-like to_replace "
   6681             "and non-None value"
   6682         )
   6683     mapping = {
   6684         col: (to_rep, value) for col, to_rep in to_replace.items()
   6685     }
   6686     return self._replace_columnwise(mapping, inplace, regex)

ValueError: Series.replace cannot use dict-like to_replace and non-None value
```

In [69]:
```
df_clip.describe()
```

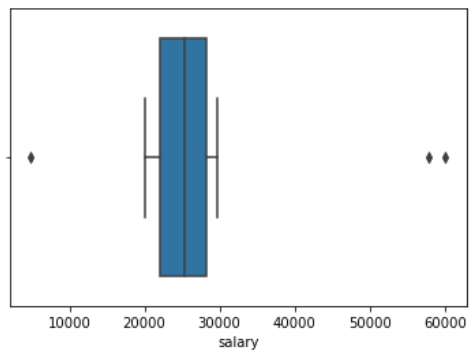
Out[69]:

	ID	salary
count	30.000000	30.000000
mean	1015.500000	26609.800000
std	8.803408	9994.181705
min	1001.000000	4780.000000
25%	1008.250000	22031.750000
50%	1015.500000	25420.000000
75%	1022.750000	28215.750000
max	1030.000000	60100.000000

In [70]:
```
sns.boxplot(df_clip['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

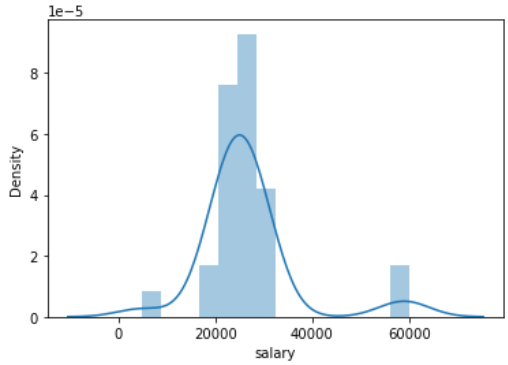
Out[70]:
```
<AxesSubplot:xlabel='salary'>
```



In [71]:
```
sns.distplot(df_clip['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[71]:
```
<AxesSubplot:xlabel='salary', ylabel='Density'>
```



Another function for clipping the outliner values

In [72]:
```
import numpy as np
df_clip['salary'] = np.where( df['salary'] >= upper_limit,
                                    upper_limit,
                                    np.where(df['salary'] <= lower_limit,
                                                lower_limit,
                                                df['salary'] ) )
```

In [73]:
```
df_clip.describe()
```

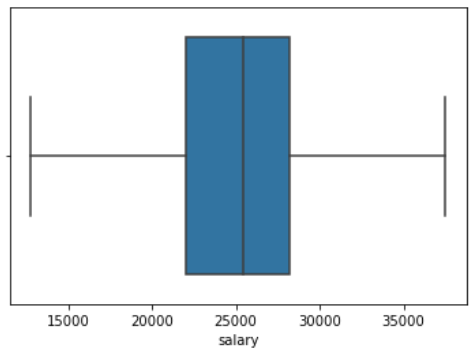
Out[73]:

	ID	salary
count	30.000000	30.000000
mean	1015.500000	25445.075000
std	8.803408	4960.786202
min	1001.000000	12755.750000
25%	1008.250000	22031.750000
50%	1015.500000	25420.000000
75%	1022.750000	28215.750000
max	1030.000000	37491.750000

In [74]:
```
sns.boxplot(df_clip['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

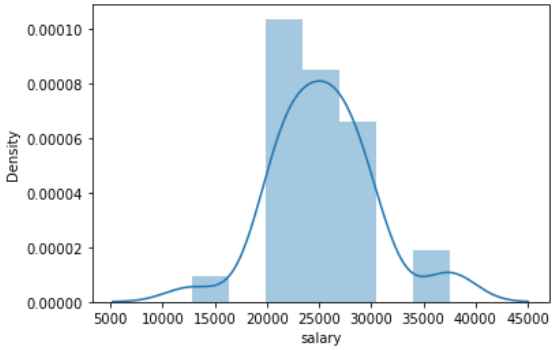
Out[74]:
```
<AxesSubplot:xlabel='salary'>
```



In [75]:
```
sns.distplot(df_clip['salary'])
```

```
/Users/jakapongtosunpul/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[75]:
```
<AxesSubplot:xlabel='salary', ylabel='Density'>
```


0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
Density
5000
10000
15000
20000
25000
30000
35000
40000
45000
salary

In [ ]: