In [1]:

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

In [2]:

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

Out[2]:

	CASENUM	ATTORNEY	CLMSEX	CLMINSUR	SEATBELT	CLMAGE	LOSS
0	5	0	0.0	1.0	0.0	50.0	34.940
1	3	1	1.0	0.0	0.0	18.0	0.891
2	66	1	0.0	1.0	0.0	5.0	0.330
3	70	0	0.0	1.0	1.0	31.0	0.037
4	96	1	0.0	1.0	0.0	30.0	0.038
...	...	...	...	...	...	...	...
1335	34100	1	0.0	1.0	0.0	NaN	0.576
1336	34110	0	1.0	1.0	0.0	46.0	3.705
1337	34113	1	1.0	1.0	0.0	39.0	0.099
1338	34145	0	1.0	0.0	0.0	8.0	3.177
1339	34153	1	1.0	1.0	0.0	30.0	0.688

1340 rows × 7 columns

In [3]:

```
df.shape
```

Out[3]:

```
(1340, 7)
```

In [4]:

```
df.info
```

Out[4]:

```
<bound method DataFrame.info of       CASENUM  ATTORNEY  CLMSEX  CLMINSUR  SEATBELT  CLM
AGE    LOSS
0           5         0     0.0       1.0       0.0    50.0  34.940
1           3         1     1.0       0.0       0.0    18.0   0.891
2          66         1     0.0       1.0       0.0     5.0   0.330
3          70         0     0.0       1.0       1.0    31.0   0.037
4          96         1     0.0       1.0       0.0    30.0   0.038
...       ...       ...     ...       ...       ...     ...     ...
1335    34100         1     0.0       1.0       0.0     NaN   0.576
1336    34110         0     1.0       1.0       0.0    46.0   3.705
1337    34113         1     1.0       1.0       0.0    39.0   0.099
1338    34145         0     1.0       0.0       0.0     8.0   3.177
1339    34153         1     1.0       1.0       0.0    30.0   0.688

[1340 rows x 7 columns]>
```

In [5]:

```
df.describe()
```

Out[5]:

	CASENUM	ATTORNEY	CLMSEX	CLMINSUR	SEATBELT	CLMAGE	LOSS
count	1340.000000	1340.000000	1328.000000	1299.000000	1292.000000	1151.000000	1340.000000
mean	11202.001493	0.488806	0.558735	0.907621	0.017028	28.414422	3.806307
std	9512.750796	0.500061	0.496725	0.289671	0.129425	20.304451	10.636903
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	4177.000000	0.000000	0.000000	1.000000	0.000000	9.000000	0.400000
50%	8756.500000	0.000000	1.000000	1.000000	0.000000	30.000000	1.069500
75%	15702.500000	1.000000	1.000000	1.000000	0.000000	43.000000	3.781500
max	34153.000000	1.000000	1.000000	1.000000	1.000000	95.000000	173.604000

In [6]:

```
df.mode()
```

Out[6]:

	CASENUM	ATTORNEY	CLMSEX	CLMINSUR	SEATBELT	CLMAGE	LOSS
0	590	0.0	1.0	1.0	0.0	10.0	0.0
1	1010	NaN	NaN	NaN	NaN	NaN	NaN
2	6330	NaN	NaN	NaN	NaN	NaN	NaN

In [7]:

```
df.median()
```

Out[7]:

```
CASENUM     8756.5000
ATTORNEY       0.0000
CLMSEX         1.0000
CLMINSUR       1.0000
SEATBELT       0.0000
CLMAGE        30.0000
LOSS           1.0695
dtype: float64
```

In [8]:

```
df.var()
```

Out[8]:

```
CASENUM     9.049243e+07
ATTORNEY    2.500613e-01
CLMSEX      2.467360e-01
CLMINSUR    8.390951e-02
SEATBELT    1.675088e-02
CLMAGE      4.122707e+02
LOSS        1.131437e+02
dtype: float64
```

In [9]:

```
df.skew()
```

Out[9]:

```
CASENUM     1.060846
ATTORNEY    0.044838
CLMSEX     -0.236845
```

```
CLMINSUR   -2.818710
SEATBELT    7.474911
CLMAGE      0.410035
LOSS        7.724874
dtype: float64
```

In [10]:

```
df.kurtosis()
```

Out[10]:

```
CASENUM      0.244294
ATTORNEY    -2.000978
CLMSEX      -1.946839
CLMINSUR     5.954291
SEATBELT    53.957825
CLMAGE      -0.745531
LOSS        79.155257
dtype: float64
```

In [11]:

```
df.columns
```

Out[11]:

```
Index(['CASENUM', 'ATTORNEY', 'CLMSEX', 'CLMINSUR', 'SEATBELT', 'CLMAGE',
       'LOSS'],
      dtype='object')
```

In [12]:

```
del df["CASENUM"]
```

In [13]:

```
df.head(7)
```

Out[13]:

	ATTORNEY	CLMSEX	CLMINSUR	SEATBELT	CLMAGE	LOSS
0	0	0.0	1.0	0.0	50.0	34.940
1	1	1.0	0.0	0.0	18.0	0.891
2	1	0.0	1.0	0.0	5.0	0.330
3	0	0.0	1.0	1.0	31.0	0.037
4	1	0.0	1.0	0.0	30.0	0.038
5	0	1.0	1.0	0.0	35.0	0.309
6	0	0.0	1.0	0.0	9.0	3.538

In [14]:

```
df.isnull().sum()
```

Out[14]:

```
ATTORNEY      0
CLMSEX       12
CLMINSUR     41
SEATBELT     48
CLMAGE      189
LOSS          0
dtype: int64
```

In [15]:

```
df['CLMSEX'].fillna(df['CLMSEX'].mode() [0], inplace=True)
df['CLMINSUR'].fillna(df['CLMINSUR'].mode() [0], inplace=True)
df['SEATBELT'].fillna(df['SEATBELT'].mode() [0], inplace=True)
```

In [16]:

```
df['CLMAGE'].fillna(df['CLMAGE'].mean(), inplace=True)
```

In [17]:

```
df.isnull().sum()
```

Out[17]:

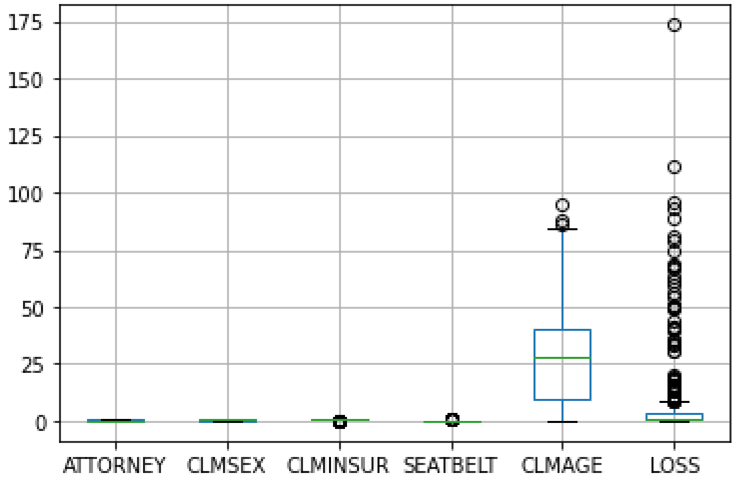
```
ATTORNEY    0
CLMSEX      0
CLMINSUR    0
SEATBELT    0
CLMAGE      0
LOSS        0
dtype: int64
```

In [18]:

```
boxplot=df.boxplot()
boxplot
```

Out[18]:

```
<AxesSubplot:>
```



In [19]:

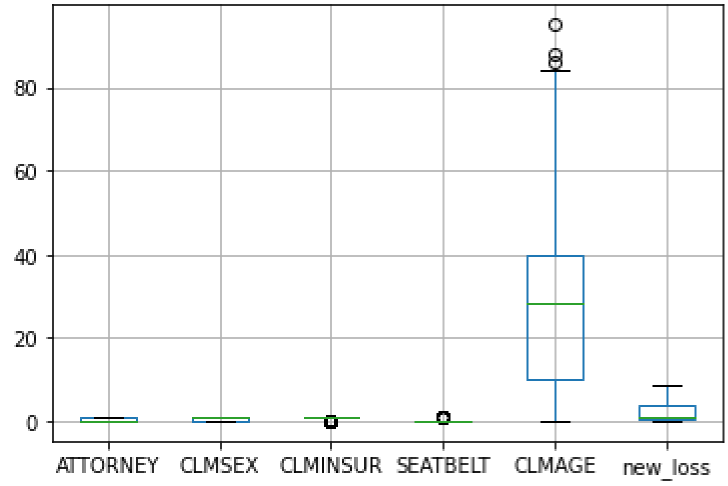
```
from feature_engine.outliers import Winsorizer
win=Winsorizer(capping_method="iqr", tail="both", fold=1.5, variables=["LOSS"])
new_loss=win.fit_transform(df[["LOSS"]])
df.insert(loc=5,column='new_loss', value=new_loss)
```

In [20]:

```
del df["LOSS"]
boxplot=df.boxplot()
boxplot
```

Out[20]:

```
<AxesSubplot:>
```



In [21]:

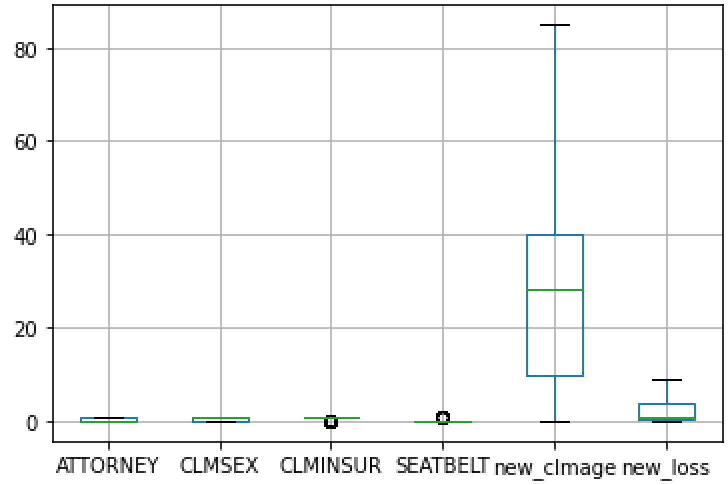
```
from feature_engine.outliers import Winsorizer
win=Winsorizer(capping_method="iqr", tail="both", fold=1.5, variables=["CLMAGE"])
new_clmage=win.fit_transform(df[["CLMAGE"]])
df.insert(loc=5,column='new_clmage', value=new_clmage)
```

In [22]:

```
del df["CLMAGE"]
boxplot=df.boxplot()
boxplot
```

Out[22]:

```
<AxesSubplot:>
```



In [23]:

```
df
```

Out[23]:

	ATTORNEY	CLMSEX	CLMINSUR	SEATBELT	new_clmage	new_loss
0	0	0.0	1.0	0.0	50.000000	8.85375
1	1	1.0	0.0	0.0	18.000000	0.89100
2	1	0.0	1.0	0.0	5.000000	0.33000
3	0	0.0	1.0	1.0	31.000000	0.03700
4	1	0.0	1.0	0.0	30.000000	0.03800

	ATTORNEY	CLMSEX	CLMINSUR	SEATBELT	new_clmage	new_loss
...	...	...	...	...	...	...
1335	1	0.0	1.0	0.0	28.414422	0.57600
1336	0	1.0	1.0	0.0	46.000000	3.70500
1337	1	1.0	1.0	0.0	39.000000	0.09900
1338	0	1.0	0.0	0.0	8.000000	3.17700
1339	1	1.0	1.0	0.0	30.000000	0.68800

1340 rows × 6 columns

In [32]:

```
y=df.iloc[:,0]
x=df.iloc[:, 1:]
```

In [33]:

```
y
```

Out[33]:

```
0       0
1       1
2       1
3       0
4       1
       ..
1335    1
1336    0
1337    1
1338    0
1339    1
Name: ATTORNEY, Length: 1340, dtype: int64
```

In [35]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test=train_test_split(x,y, test_size=0.25)
```

In [37]:

```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
x_train=scaler.fit_transform(x_train)
x_test=scaler.transform(x_test)
```

In [38]:

```
from sklearn.linear_model import LogisticRegression
clf=LogisticRegression(random_state=0)
clf.fit(x_train, y_train)
```

Out[38]:

```
LogisticRegression(random_state=0)
```

In [39]:

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

Out[39]:

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

In [43]:

```
y_test
```

Out[43]:

```
179     0
171     0
172     0
338     1
257     0
       ..
828     0
955     1
1211    1
369     1
784     1
Name: ATTORNEY, Length: 335, dtype: int64
```

In [44]:

```
from sklearn.metrics import confusion_matrix, accuracy_score
CM=confusion_matrix(y_test, y_pred)
accuracy=accuracy_score(y_test, y_pred)
```

In [45]:

```
CM
```

Out[45]:

```
array([[ 49, 123],
       [ 15, 148]], dtype=int64)
```

In [46]:

```
accuracy
```

Out[46]:

```
0.5880597014925373
```

In [ ]: