In [20]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

In [2]:
```
bankdata = pd.read_csv('D:\\24 - Machine_Learning\\download files\\bill_authentication.csv')
bankdata
```

Out[2]:

	Variance	Skewness	Curtosis	Entropy	Class
0	3.62160	8.66610	-2.8073	-0.44699	0
1	4.54590	8.16740	-2.4586	-1.46210	0
2	3.86600	-2.63830	1.9242	0.10645	0
3	3.45660	9.52280	-4.0112	-3.59440	0
4	0.32924	-4.45520	4.5718	-0.98880	0
...	...	...	...	...	...
1367	0.40614	1.34920	-1.4501	-0.55949	1
1368	-1.38870	-4.87730	6.4774	0.34179	1
1369	-3.75030	-13.45860	17.5932	-2.77710	1
1370	-3.56370	-8.38270	12.3930	-1.28230	1
1371	-2.54190	-0.65804	2.6842	1.19520	1

1372 rows × 5 columns

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

Out[3]:

	Variance	Skewness	Curtosis	Entropy	Class
count	1372.000000	1372.000000	1372.000000	1372.000000	1372.000000
mean	0.433735	1.922353	1.397627	-1.191657	0.444606
std	2.842763	5.869047	4.310030	2.101013	0.497103
min	-7.042100	-13.773100	-5.286100	-8.548200	0.000000
25%	-1.773000	-1.708200	-1.574975	-2.413450	0.000000
50%	0.496180	2.319650	0.616630	-0.586650	0.000000
75%	2.821475	6.814625	3.179250	0.394810	1.000000
max	6.824800	12.951600	17.927400	2.449500	1.000000

In [4]:
```
bankdata.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1372 entries, 0 to 1371
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Variance  1372 non-null   float64
 1   Skewness  1372 non-null   float64
 2   Curtosis  1372 non-null   float64
 3   Entropy   1372 non-null   float64
 4   Class     1372 non-null   int64  
dtypes: float64(4), int64(1)
memory usage: 53.7 KB
```

In [5]:
```
bankdata.isnull()
```

Out[5]:

	Variance	Skewness	Curtosis	Entropy	Class
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
...	...	...	...	...	...
1367	False	False	False	False	False
1368	False	False	False	False	False
1369	False	False	False	False	False
1370	False	False	False	False	False
1371	False	False	False	False	False

1372 rows × 5 columns

In [6]:
```
sns.heatmap(bankdata.isnull())
```

Out[6]:
```
<AxesSubplot:>
```

In [7]:
```
bankdata.head(10)
```

Out[7]:

	Variance	Skewness	Curtosis	Entropy	Class
0	3.62160	8.6661	-2.80730	-0.44699	0
1	4.54590	8.1674	-2.45860	-1.46210	0
2	3.86600	-2.6383	1.92420	0.10645	0
3	3.45660	9.5228	-4.01120	-3.59440	0
4	0.32924	-4.4552	4.57180	-0.98880	0
5	4.36840	9.6718	-3.96060	-3.16250	0
6	3.59120	3.0129	0.72888	0.56421	0
7	2.09220	-6.8100	8.46360	-0.60216	0
8	3.20320	5.7588	-0.75345	-0.61251	0
9	1.53560	9.1772	-2.27180	-0.73535	0

In [10]:
```
X = bankdata.drop(['Class'],axis=1)
X
```

Out[10]:

	Variance	Skewness	Curtosis	Entropy
0	3.62160	8.66610	-2.8073	-0.44699
1	4.54590	8.16740	-2.4586	-1.46210
2	3.86600	-2.63830	1.9242	0.10645
3	3.45660	9.52280	-4.0112	-3.59440
4	0.32924	-4.45520	4.5718	-0.98880
...	...	...	...	...
1367	0.40614	1.34920	-1.4501	-0.55949
1368	-1.38870	-4.87730	6.4774	0.34179
1369	-3.75030	-13.45860	17.5932	-2.77710
1370	-3.56370	-8.38270	12.3930	-1.28230
1371	-2.54190	-0.65804	2.6842	1.19520

1372 rows × 4 columns

In [12]:
```
Y = bankdata['Class']
Y
```

Out[12]:
```
0       0
1       0
2       0
3       0
4       0
       ..
1367    1
1368    1
1369    1
1370    1
1371    1
Name: Class, Length: 1372, dtype: int64
```

In [13]:
```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.20)
```

In [16]:
```
svclassifier = SVC(kernel = 'linear')
svclassifier.fit(X_train, Y_train)
```

Out[16]:
```
SVC(kernel='linear')
```

In [17]:
```
Y_pred = svclassifier.predict(X_test)
Y_pred
```

Out[17]:
```
array([0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0,
       0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1,
       1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1,
       1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0,
       0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0,
       0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1,
       0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0,
       0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0,
       0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0], dtype=int64)
```

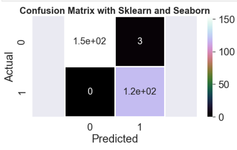
In [19]:
```
svclassifier.score(X_test, Y_test)
```

Out[19]:
```
0.9890909090909091
```

In [21]:
```
cf = confusion_matrix(Y_test, Y_pred)
cf
```

Out[21]:
```
array([[154,   3],
       [  0, 118]], dtype=int64)
```

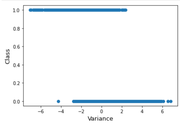
In [22]:
```
with plt.style.context('seaborn'):
    plt.figure(figsize=(8,4))
    sns.set(font_scale=2)
    sns.heatmap(cf, annot = True, square=True, annot_kws={'size':20}, linewidth=3, cmap='cubehelix')
    plt.xlabel('Predicted');
    plt.ylabel('Actual')
    plt.axis('equal');
    plt.title('Confusion Matrix with Sklearn and Seaborn', fontweight='bold', fontsize=20)
    plt.show()
```



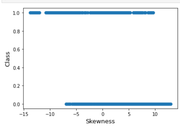
In [23]:
```
# CLASSIFICATION REPORT
print(classification_report(Y_test, Y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      0.98      0.99       157
           1       0.98      1.00      0.99       118

    accuracy                           0.99       275
   macro avg       0.99      0.99      0.99       275
weighted avg       0.99      0.99      0.99       275
```

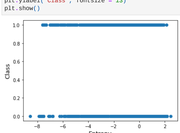
In [25]:
```
plt.scatter(bankdata['Variance'], Y)
plt.xlabel('Variance', fontsize = 13)
plt.ylabel('Class', fontsize = 13)
plt.show()
```



In [26]:
```
plt.scatter(bankdata['Skewness'], Y)
plt.xlabel('Skewness', fontsize = 13)
plt.ylabel('Class', fontsize = 13)
plt.show()
```



In [27]:
```
plt.scatter(bankdata['Entropy'], Y)
plt.xlabel('Entropy', fontsize = 13)
plt.ylabel('Class', fontsize = 13)
plt.show()
```



In [28]:
```
plt.scatter(bankdata['Curtosis'], Y)
plt.xlabel('Curtosis', fontsize = 13)
plt.ylabel('Class', fontsize = 13)
plt.show()
```