```python
In [15]: import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         from sklearn import model_selection
         from sklearn.svm import SVC
         from sklearn.ensemble import BaggingClassifier
         from sklearn.model_selection import train_test_split
```

```python
In [16]: bankdata = pd.read_csv("C:/Users/DEEP SUMAN/Desktop/Lab Prg Ready/bill_authentication.csv")
```

```python
In [17]: bankdata.shape
```

```
Out[17]: (1372, 5)
```

```python
In [18]: bankdata.head(10)
```

Out[18]:

	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

```python
In [19]: X = bankdata.drop('Class', axis=1)
         y = bankdata['Class']
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20)
```

```python
In [20]: # PROCESS 1
         model = BaggingClassifier(SVC())
```

```python
In [21]: kfold=model_selection.KFold(n_splits=10)
         results = model_selection.cross_val_score(model,X_train, y_train, cv=kfold)
```

```python
In [22]: print("Results:", results)
         print("Mean Results:", results.mean())
```

```
Results: [1.          0.99090909 1.          1.          1.          1.
 1.          1.          0.98165138 1.         ]
Mean Results: 0.997256046705588
```

```python
In [24]: #PROCESS 2
         model = BaggingClassifier(SVC())
         model.fit(X_train, y_train)
```

Out[24]:

▸ **BaggingClassifier**

   ▸ **estimator: SVC**

      ▸ SVC

```python
In [25]: y_pred = model.predict(X_test)
```

```python
In [26]: model.score(X_test, y_test)
```

```
Out[26]: 1.0
```

```python
In [27]: from sklearn.metrics import classification_report, confusion_matrix
         print(confusion_matrix(y_test,y_pred))
         print(classification_report(y_test,y_pred))
```

```
[[144   0]
 [  0 131]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       144
           1       1.00      1.00      1.00       131

    accuracy                           1.00       275
   macro avg       1.00      1.00      1.00       275
weighted avg       1.00      1.00      1.00       275
```