Support Vector Machine using Bill Authentication Dataset

Part B Assignment 6

Name : Devashish Prasad

Roll No. 43320

A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labeled training data for each category, they're able to categorize new text.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

```
svm_data = pd.read_csv("bill_authentication.csv")
```

```
svm_data.shape
```

```
(1372, 5)
```

```
svm_data.head()
```

	Variance	Skewness	Curtosis	Entropy	Class
0	3.62160	8.6661	-2.8073	-0.44699	0
1	4.54590	8.1674	-2.4586	-1.46210	0
2	3.86600	-2.6383	1.9242	0.10645	0
3	3.45660	9.5228	-4.0112	-3.59440	0
4	0.32924	-4.4552	4.5718	-0.98880	0

Our task is to predict whether a bank currency note is **authentic or not** based upon four attributes of the note i.e. skewness of the wavelet transformed image, variance of the image, entropy of the image, and curtosis of the image. This is a **binary classification problem and I will use SVM algorithm** to solve this problem. The rest of the section consists of standard machine learning steps.

Preparing Features and Target Variable

```
X = svm_data.drop('Class', axis=1)
y = svm_data['Class']
```

Splitting the Dataset for training and testing

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20)
```

```
print(X_train.shape)
print(X_test.shape)
```

```
(1097, 4)
(275, 4)
```

Apply SVM using the linear kernel

```
from sklearn.svm import SVC
svclassifier = SVC(kernel='linear')
svclassifier.fit(X_train, y_train)
```

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='linear',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

Prediction

```
y_pred = svclassifier.predict(X_test)
```

Confusion Matrix

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

```

```
[[138   3]
 [  0 134]]
```

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      0.98      0.99       141
           1       0.98      1.00      0.99       134

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