import pandas as pd  
  
from sklearn import svm  
  
from sklearn.datasets import load\_breast\_cancer  
  
from sklearn.model\_selection import train\_test\_split  
from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, classification\_report

data = load\_breast\_cancer()  
  
x = pd.DataFrame(data.data)  
y = pd.Series(data.target)

x

0 1 2 3 4 5 6 7 8 \  
0 17.99 10.38 122.80 1001.0 0.11840 0.27760 0.30010 0.14710 0.2419   
1 20.57 17.77 132.90 1326.0 0.08474 0.07864 0.08690 0.07017 0.1812   
2 19.69 21.25 130.00 1203.0 0.10960 0.15990 0.19740 0.12790 0.2069   
3 11.42 20.38 77.58 386.1 0.14250 0.28390 0.24140 0.10520 0.2597   
4 20.29 14.34 135.10 1297.0 0.10030 0.13280 0.19800 0.10430 0.1809   
.. ... ... ... ... ... ... ... ... ...   
564 21.56 22.39 142.00 1479.0 0.11100 0.11590 0.24390 0.13890 0.1726   
565 20.13 28.25 131.20 1261.0 0.09780 0.10340 0.14400 0.09791 0.1752   
566 16.60 28.08 108.30 858.1 0.08455 0.10230 0.09251 0.05302 0.1590   
567 20.60 29.33 140.10 1265.0 0.11780 0.27700 0.35140 0.15200 0.2397   
568 7.76 24.54 47.92 181.0 0.05263 0.04362 0.00000 0.00000 0.1587   
  
 9 ... 20 21 22 23 24 25 26 \  
0 0.07871 ... 25.380 17.33 184.60 2019.0 0.16220 0.66560 0.7119   
1 0.05667 ... 24.990 23.41 158.80 1956.0 0.12380 0.18660 0.2416   
2 0.05999 ... 23.570 25.53 152.50 1709.0 0.14440 0.42450 0.4504   
3 0.09744 ... 14.910 26.50 98.87 567.7 0.20980 0.86630 0.6869   
4 0.05883 ... 22.540 16.67 152.20 1575.0 0.13740 0.20500 0.4000   
.. ... ... ... ... ... ... ... ... ...   
564 0.05623 ... 25.450 26.40 166.10 2027.0 0.14100 0.21130 0.4107   
565 0.05533 ... 23.690 38.25 155.00 1731.0 0.11660 0.19220 0.3215   
566 0.05648 ... 18.980 34.12 126.70 1124.0 0.11390 0.30940 0.3403   
567 0.07016 ... 25.740 39.42 184.60 1821.0 0.16500 0.86810 0.9387   
568 0.05884 ... 9.456 30.37 59.16 268.6 0.08996 0.06444 0.0000   
  
 27 28 29   
0 0.2654 0.4601 0.11890   
1 0.1860 0.2750 0.08902   
2 0.2430 0.3613 0.08758   
3 0.2575 0.6638 0.17300   
4 0.1625 0.2364 0.07678   
.. ... ... ...   
564 0.2216 0.2060 0.07115   
565 0.1628 0.2572 0.06637   
566 0.1418 0.2218 0.07820   
567 0.2650 0.4087 0.12400   
568 0.0000 0.2871 0.07039   
  
[569 rows x 30 columns]

y

0 0  
1 0  
2 0  
3 0  
4 0  
 ..  
564 0  
565 0  
566 0  
567 0  
568 1  
Length: 569, dtype: int64

# Using 70-30 Split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.3, random\_state=42)

# Create a Classifier  
cls = svm.SVC(kernel="linear")  
  
# Train the model  
cls.fit(x\_train,y\_train)  
  
# Predict the response  
y\_pred = cls.predict(x\_test)

# Accuracy  
print("Acuracy: %.2f%%" % (accuracy\_score(y\_test, y\_pred) \* 100))  
# Precision score  
print("Precision: %.2f" % precision\_score(y\_test, y\_pred))  
# Recall score  
print("Recall: %.2f" % recall\_score(y\_test, y\_pred))  
  
# Classification Report  
print(classification\_report(y\_test, y\_pred))

Acuracy: 96.49%  
Precision: 0.96  
Recall: 0.98  
 precision recall f1-score support  
  
 0 0.97 0.94 0.95 63  
 1 0.96 0.98 0.97 108  
  
 accuracy 0.96 171  
 macro avg 0.97 0.96 0.96 171  
weighted avg 0.96 0.96 0.96 171

# Using 80-20 Split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state=42)

# Create a Classifier  
cls = svm.SVC(kernel="linear")  
  
# Train the model  
cls.fit(x\_train,y\_train)  
  
# Predict the response  
y\_pred = cls.predict(x\_test)

# Accuracy  
print("Acuracy: %.2f%%" % (accuracy\_score(y\_test, y\_pred) \* 100))  
# Precision score  
print("Precision: %.2f" % precision\_score(y\_test, y\_pred))  
# Recall score  
print("Recall: %.2f" % recall\_score(y\_test, y\_pred))  
  
# Classification Report  
print(classification\_report(y\_test, y\_pred))

Acuracy: 95.61%  
Precision: 0.95  
Recall: 0.99  
 precision recall f1-score support  
  
 0 0.97 0.91 0.94 43  
 1 0.95 0.99 0.97 71  
  
 accuracy 0.96 114  
 macro avg 0.96 0.95 0.95 114  
weighted avg 0.96 0.96 0.96 114