CS353 ML Lab 7

Name: K V Sumanth Reddy

Roll No: 181CO225

Batch: Section 2

Date: 23/03/2021

Q: Write a program in python to implement and demostrate **logistic regression** for a sample training dataset. Compute the accuracy of the classifier.

Dataset Used: Breast Cancer Dataset

Importing Libraries and Dataset

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

from sklearn.datasets import load_breast_cancer
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler

from sklearn.metrics import plot_confusion_matrix,precision_score
from sklearn.metrics import mean_squared_error
from sklearn.metrics import accuracy_score, classification_report
```

Data Preprocessing

```
dataset = load_breast_cancer()
data = pd.DataFrame(dataset.data, columns=[dataset.feature_names])
data['Target'] = pd.Series(data=dataset.target, index=data.index)

x,y = load_breast_cancer(return_X_y=True)
data.sample(5)
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean conc poir
54	15.10	22.02	97.26	712.8	0.09056	0.07081	0.05253	0.0
190	14.22	23.12	94.37	609.9	0.10750	0.24130	0.19810	0.0
481	13.90	19.24	88.73	602.9	0.07991	0.05326	0.02995	0.0
485	12.45	16.41	82.85	476.7	0.09514	0.15110	0.15440	0.0
402	12.96	18.29	84.18	525.2	0.07351	0.07899	0.04057	0.0

```
print("Label: ", dataset.target_names)
```

```
Label:  ['malignant' 'benign']
```

```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size
print('Training dataset size:\nx_train -', len(x_train), '\ny_train
print('Testing dataset size:\nx_test -', len(x_test), '\ny_test -',
```

```
Training dataset size:
x_train - 398
y_train - 398

Testing dataset size:
x_test - 171
y_test - 171
```

```
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

```
model = LogisticRegression()
model.fit(x_train,y_train)
y_pred = model.predict(x_test)
```

Results

```
print('------------------------------')
print('Accuracy:                {:f}'.format(accuracy_score(y_test,
print('Precision:               {:f}'.format(precision_score(y_test,
print('Mean Squared Error:      {:f}'.format(mean_squared_error(y_te
print('------------------------------')
```

```
------------------------------
Accuracy:                96.491228
Precision:               94.444444
Mean Squared Error:      3.508772
------------------------------
```

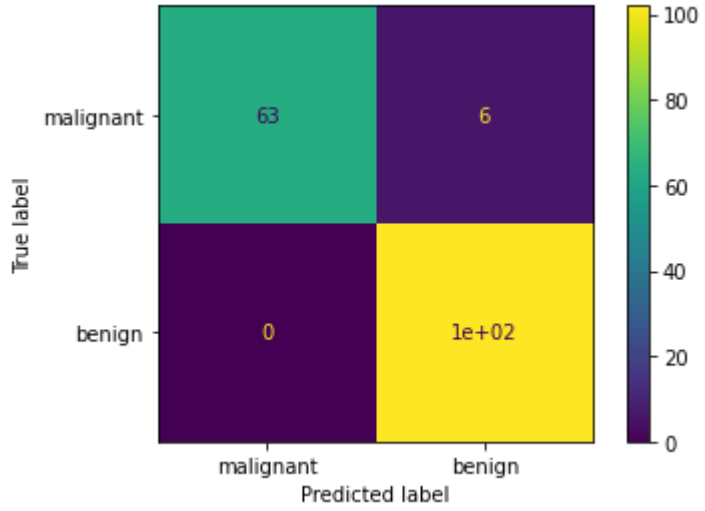
```
print(classification_report(y_test,y_pred))
```

```
                precision    recall  f1-score   support

             0       1.00      0.91      0.95        69
             1       0.94      1.00      0.97       102

      accuracy                           0.96       171
     macro avg       0.97      0.96      0.96       171
  weighted avg       0.97      0.96      0.96       171
```

```
plot_confusion_matrix(model, x_test, y_test, display_labels=dataset.
plt.show()
```



✓ 0s completed at 15:47