Haberman Data

Supervised learning

Nividha Jadeja (340)

1. **Introduction**:

   The dataset contains cases from a study that was conducted between 1958 and 1970 at the University of Chicago's Billings Hospital on the survival of patients who had undergone surgery for breast cancer.

2. **Dataset**:

   The data is in a structured format in a CSV file.

A	B	C	D
Age_of_patient	Patient's_year_of_operation	Number_of_nodes	Survival_status
30	64	1	1
30	62	3	1
30	65	0	1
31	59	2	1
31	65	4	1
33	58	10	1
33	60	0	1
34	59	0	2
34	66	9	2
34	58	30	1
34	60	1	1
34	61	10	1
34	67	7	1
34	60	0	1
35	64	13	1

3. **Data Preparation**:

   The following steps have been ensured for data preparation:

   - Querying the data using Pandas to check if all the data has been imported for processing.
   - Formatting of data for survivors more than five years as 1 and those who didn't survive as 0 using toReplace().

```
df.Survival_status = df.Survival_status.replace(to_replace = [2], regex = True, value = [0])
df.head(200)
```

	Age_of_patient	Patient's_year_of_operation	Number_of_nodes	Survival_status
0	30	64	1	1
1	30	62	3	1
2	30	65	0	1
3	31	59	2	1
4	31	65	4	1
...	...	...	...	...
195	56	67	0	1
196	56	60	0	1
197	57	61	5	0

4. **Feature Engineering**:

The main types of features are- Categorical, Continuous and Derived features. As it is a Binary Classification problem, it is a categorical feature

```
print('Target variables  : ', df['Survival_status'])

(unique, counts) = np.unique(df['Survival_status'], return_counts=True)

print('Unique values of the target variable', unique)
print('Counts of the target variable :', counts)
```

```
Target variables  :  0      1
1      1
2      1
3      1
4      1
      ..
301    1
302    1
303    1
304    0
305    0
Name: Survival_status, Length: 306, dtype: int64
Unique values of the target variable [0 1]
Counts of the target variable : [ 81 225]
```

```
# Putting feature variable to X
X = df.iloc[:,[0,1,2]]
# Putting response variable to y
y = df.iloc[:, 3]
```

5. **Data Modelling**:

Model Scaling

```
from sklearn.preprocessing import StandardScaler
standardizer = StandardScaler()
X = standardizer.fit_transform(X)
```

Splitting the dataset

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y , test_size=0.25, random_state=0)
```

Logistic regression model is used for Binary Classification

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X_train, y_train)
```

```
LogisticRegression()
```

6. **Performance Evaluation**:

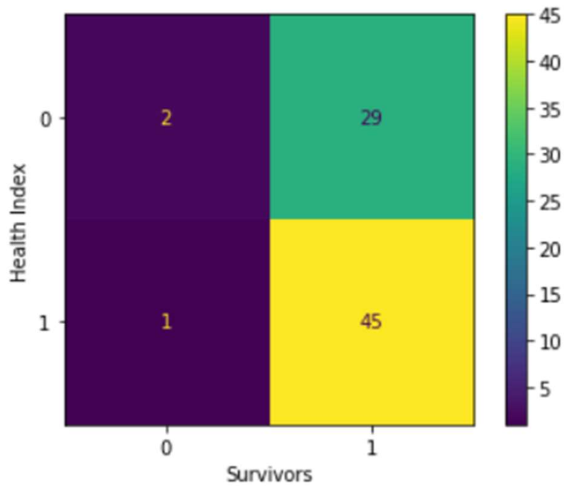
Confusion matrix

```
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, predictions)

TN, FP, FN, TP = confusion_matrix(y_test, predictions).ravel()

print('True Positive(TP)  = ', TP)
print('False Positive(FP) = ', FP)
print('True Negative(TN)  = ', TN)
print('False Negative(FN) = ', FN)

accuracy =  (TP+TN) /(TP+FP+TN+FN)

print('Accuracy of the binary classification = {:0.3f}'.format(accuracy))
```

```
True Positive(TP)  =  45
False Positive(FP) =  29
True Negative(TN)  =  2
False Negative(FN) =  1
Accuracy of the binary classification = 0.610
```

Graphical representation

```
from sklearn.metrics import plot_confusion_matrix
plot_confusion_matrix(model, X_test, y_test)
plt.xlabel("Survivors")
plt.ylabel("Health Index")
plt.show()
```



7. **References**:

- https://archive.ics.uci.edu/ml/datasets/Haberman%27s+Survival
- https://stackoverflow.com/questions/40901770/is-there-a-simple-way-to-change-a-column-of-yes-no-to-1-0-in-a-pandas-dataframe
- https://stackabuse.com/classification-in-python-with-scikit-learn-and-pandas/
- https://scikit-learn.org/stable/modules/generated/sklearn.metrics.ConfusionMatrixDisplay.html#sklearn.metrics.ConfusionMatrixDisplay