Sonar Rock And Mine Prediction

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

#loading the dataset to a pandas Dataframe
sonar_data = pd.read_csv('D:\project AI\Copy of sonar data.csv', header=None)
sonar_data.head()
# number of rows and columns
sonar_data.shape
sonar_data.describe()  #describe --> statistical measures of the data
```

```
sonar_data[60].value_counts()
sonar_data.groupby(60).mean()
# separating data and Labels
X = sonar_data.drop(columns=60, axis=1)
Y = sonar_data[60]
print(X)
print(Y)
# Training and Test data
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.1, stratify=Y, random_state=1)
print(X.shape, X_train.shape, X_test.shape)
```

```
print(X_train)
print(Y_train)
```

```
#Model Training --> Logistic Regression
model = LogisticRegression()
#training the Logistic Regression model with training data
model.fit(X_train, Y_train)
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

```
#accuracy on training data
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
print('Accuracy on training data : ', training_data_accuracy)
```

```
#accuracy on test data
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print('Accuracy on test data : ', test_data_accuracy)
```

```
# Making a Predictive System
input_data =
(0.0307,0.0523,0.0653,0.0521,0.0611,0.0577,0.0665,0.0664,0.1460,0.2792,0.3877,0.4992,0.4981,0.4972,0.5607,0.
7339,0.8230,0.9173,0.9975,0.9911,0.8240,0.6498,0.5980,0.4862,0.3150,0.1543,0.0989,0.0284,0.1008,0.2636,0.26
94,0.2930,0.2925,0.3998,0.3660,0.3172,0.4609,0.4374,0.1820,0.3376,0.6202,0.4448,0.1863,0.1420,0.0589,0.0576,
0.0672,0.0269,0.0245,0.0190,0.0063,0.0321,0.0189,0.0137,0.0277,0.0152,0.0052,0.0121,0.0124,0.0055)
```

```
# changing the input_data to a numpy array
input_data_as_numpy_array = np.asarray(input_data)
```

```
# reshape the np array as we are predicting for one instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)
```

```
prediction = model.predict(input_data_reshaped)
print(prediction)
```

```
if (prediction[0]=='R'):
  print('The object is a Rock')
else:
  print('The object is a mine')
```