Logistic Regression

In [1]:
```
import numpy as np
import pandas as pd
import xlrd
from sklearn import svm
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
import sys
!{sys.executable} -m pip install openpyxl
import warnings
warnings.filterwarnings('ignore')
warnings.simplefilter('ignore')
```

```
Requirement already satisfied: openpyxl in c:\users\okina\anaconda3\lib\site-packages (3.0.10)
Requirement already satisfied: et_xmlfile in c:\users\okina\anaconda3\lib\site-packages (from openpyxl)
(1.1.0)
```

In [2]:
```
new_df = pd.read_excel('Immunotherapy.xlsx')
```

In [3]:
```
def calc_error(X,Y, classifier):
    Y_pred = classifier.predict(X)
    accuracy = accuracy_score(Y, Y_pred)
    error = 1 - accuracy

    return error,accuracy
```

In [4]:
```
X = np.asarray(new_df[['sex', 'age', 'Time', 'Number_of_Warts', 'Type', 'Area',
       'induration_diameter']])

Y = np.asarray(new_df[['Result_of_Treatment']])

X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2, random_state = 5)

Cs = [0.1, 1, 10, 100, 1000]

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.4305555555555556
Accuracy: 0.5694444444444444
Cross Validation Score: 0.4861111111111111
C = 1
Training error: 0.3055555555555556
Accuracy: 0.6944444444444444
Cross Validation Score: 0.625
C = 10
Training error: 0.29166666666666663
Accuracy: 0.7083333333333334
Cross Validation Score: 0.6805555555555557
C = 100
Training error: 0.29166666666666663
Accuracy: 0.7083333333333334
Cross Validation Score: 0.7222222222222223
C = 1000
Training error: 0.29166666666666663
Accuracy: 0.7083333333333334
Cross Validation Score: 0.7222222222222223
```

In [5]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2, random_state = 5)

Cs = [0.1, 1, 10, 100, 1000]

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.4305555555555556
Accuracy: 0.5694444444444444
Cross Validation Score: 0.4861111111111111
C = 1
Training error: 0.3055555555555556
Accuracy: 0.6944444444444444
Cross Validation Score: 0.625
C = 10
Training error: 0.29166666666666663
Accuracy: 0.7083333333333334
Cross Validation Score: 0.6805555555555557
C = 100
Training error: 0.29166666666666663
Accuracy: 0.7083333333333334
Cross Validation Score: 0.7222222222222223
C = 1000
Training error: 0.29166666666666663
Accuracy: 0.7083333333333334
Cross Validation Score: 0.7222222222222223
```

In [6]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2, random_state = 5)

Cs = [0.1, 1, 10, 100, 1000]

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.4305555555555556
Accuracy: 0.5694444444444444
Cross Validation Score: 0.4861111111111111
C = 1
Training error: 0.3055555555555556
Accuracy: 0.6944444444444444
Cross Validation Score: 0.625
C = 10
Training error: 0.29166666666666663
Accuracy: 0.7083333333333334
Cross Validation Score: 0.6805555555555557
C = 100
Training error: 0.29166666666666663
Accuracy: 0.7083333333333334
Cross Validation Score: 0.7222222222222223
C = 1000
Training error: 0.29166666666666663
Accuracy: 0.7083333333333334
Cross Validation Score: 0.7222222222222223
```

50/50 Train/Test Split

In [7]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.5, random_state = 5)

Cs = [0.1, 1, 10, 100, 1000]

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.37777777777777777
Accuracy: 0.6222222222222222
Cross Validation Score: 0.5777777777777778
C = 1
Training error: 0.2666666666666667
Accuracy: 0.7333333333333333
Cross Validation Score: 0.6888888888888888
C = 10
Training error: 0.1333333333333333
Accuracy: 0.8666666666666667
Cross Validation Score: 0.7555555555555555
C = 100
Training error: 0.06666666666666665
Accuracy: 0.9333333333333333
Cross Validation Score: 0.8444444444444444
C = 1000
Training error: 0.06666666666666665
Accuracy: 0.9333333333333333
Cross Validation Score: 0.8444444444444444
```

In [8]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.5, random_state = 5)

Cs = [0.1, 1, 10, 100, 1000]

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.37777777777777777
Accuracy: 0.6222222222222222
Cross Validation Score: 0.5777777777777778
C = 1
Training error: 0.2666666666666667
Accuracy: 0.7333333333333333
Cross Validation Score: 0.6888888888888888
C = 10
Training error: 0.1333333333333333
Accuracy: 0.8666666666666667
Cross Validation Score: 0.7555555555555555
C = 100
Training error: 0.06666666666666665
Accuracy: 0.9333333333333333
Cross Validation Score: 0.8444444444444444
C = 1000
Training error: 0.06666666666666665
Accuracy: 0.9333333333333333
Cross Validation Score: 0.8444444444444444
```

In [9]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.5, random_state = 5)

Cs = [0.1, 1, 10, 100, 1000]

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.37777777777777777
Accuracy: 0.6222222222222222
Cross Validation Score: 0.5777777777777778
C = 1
Training error: 0.2666666666666667
Accuracy: 0.7333333333333333
Cross Validation Score: 0.6888888888888888
C = 10
Training error: 0.1333333333333333
Accuracy: 0.8666666666666667
Cross Validation Score: 0.7555555555555555
C = 100
Training error: 0.06666666666666665
Accuracy: 0.9333333333333333
Cross Validation Score: 0.8444444444444444
C = 1000
Training error: 0.06666666666666665
Accuracy: 0.9333333333333333
Cross Validation Score: 0.8444444444444444
```

In [10]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.8, random_state = 5)

Cs = [0.1, 1, 10, 100, 1000]

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.2777777777777778
Accuracy: 0.7222222222222222
Cross Validation Score: 0.5555555555555556
C = 1
Training error: 0.2777777777777778
Accuracy: 0.7222222222222222
Cross Validation Score: 0.6666666666666666
C = 10
Training error: 0.16666666666666663
Accuracy: 0.8333333333333334
Cross Validation Score: 0.6666666666666666
C = 100
Training error: 0.0
Accuracy: 1.0
Cross Validation Score: 0.611111111111111
C = 1000
Training error: 0.0
Accuracy: 1.0
Cross Validation Score: 0.5555555555555556
```

In [11]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.8, random_state = 5)

Cs = [0.1, 1, 10, 100, 1000]

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.2777777777777778
Accuracy: 0.7222222222222222
Cross Validation Score: 0.5555555555555556
C = 1
Training error: 0.2777777777777778
Accuracy: 0.7222222222222222
Cross Validation Score: 0.6666666666666666
C = 10
Training error: 0.16666666666666663
Accuracy: 0.8333333333333334
Cross Validation Score: 0.6666666666666666
C = 100
Training error: 0.0
Accuracy: 1.0
Cross Validation Score: 0.611111111111111
C = 1000
Training error: 0.0
Accuracy: 1.0
Cross Validation Score: 0.5555555555555556
```

In [12]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.8, random_state = 5)

Cs = [0.1, 1, 10, 100, 1000]

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.2777777777777778
Accuracy: 0.7222222222222222
Cross Validation Score: 0.5555555555555556
C = 1
Training error: 0.2777777777777778
Accuracy: 0.7222222222222222
Cross Validation Score: 0.6666666666666666
C = 10
Training error: 0.16666666666666663
Accuracy: 0.8333333333333334
Cross Validation Score: 0.6666666666666666
C = 100
Training error: 0.0
Accuracy: 1.0
Cross Validation Score: 0.611111111111111
C = 1000
Training error: 0.0
Accuracy: 1.0
Cross Validation Score: 0.5555555555555556
```

Raisins Dataset

In [13]:
```
new_df2 = pd.read_excel('Raisin_Dataset.xlsx')
```

In [14]:
```
X2 = np.asarray(new_df2[['Area', 'MajorAxisLength', 'MinorAxisLength', 'Eccentricity',
       'ConvexArea', 'Extent', 'Perimeter',]])

Y2 = np.asarray(new_df2[['Class']])

Cs = [0.1, 1, 10, 100, 1000]

X_train, X_test, Y_train, Y_test = train_test_split(X2,Y2, test_size = 0.2, random_state = 5)

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C)
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.85
C = 1
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.85
C = 10
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.85
C = 100
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.85
C = 1000
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.85
```

In [15]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X2,Y2, test_size = 0.2, random_state = 5)

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
```

```
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.8486111111111111
C = 1
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.8458333333333333
C = 10
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.8458333333333333
C = 100
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.8472222222222223
C = 1000
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.8541666666666666
```

In [16]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X2,Y2, test_size = 0.2, random_state = 5)

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.8486111111111111
C = 1
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.8458333333333333
C = 10
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.8458333333333333
C = 100
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.8472222222222223
C = 1000
Training error: 0.14583333333333337
Accuracy: 0.8541666666666666
Cross Validation Score: 0.8541666666666666
```

50/50

In [17]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X2,Y2, test_size = 0.5, random_state = 5)

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
```

```
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
C = 1
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
C = 10
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
C = 100
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
C = 1000
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
```

In [18]:
```
for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
C = 1
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
C = 10
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
C = 100
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
C = 1000
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
```

In [19]:
```
for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
C = 1
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
C = 10
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
C = 100
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
C = 1000
Training error: 0.1466666666666666
Accuracy: 0.8533333333333334
Cross Validation Score: 0.8577777777777778
```

20/80

In [20]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X2,Y2, test_size = 0.8, random_state = 5)

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
C = 1
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
C = 10
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
C = 100
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
C = 1000
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
```

In [21]:
```
for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
C = 1
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
C = 10
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
C = 100
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
C = 1000
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
```

In [22]:
```
for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
C = 1
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
C = 10
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
C = 100
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
C = 1000
Training error: 0.13888888888888884
Accuracy: 0.8611111111111112
Cross Validation Score: 0.8555555555555555
```

Energy Dataset

In [23]:
```
new_df3 = pd.read_excel('ENB2012_data.xlsx')
```

In [24]:
```
new_df3.columns = ['Compactness', 'Surface Area', 'Wall Area', 'Roof Area', 'Heigh', 'Orientation', 'Gla
new_df3 = new_df3.drop('Heating Load', axis = 1)
```

In [25]:
```
new_df3['Cooling Load'] = np.where(new_df3['Cooling Load'] > 22.08,1,-1)
```

In [26]:
```
X3 = np.asarray(new_df3[['Compactness', 'Surface Area', 'Wall Area', 'Roof Area', 'Heigh', 'Orientation'

Y3 = np.asarray(new_df3[['Cooling Load']])

X_train, X_test, Y_train, Y_test = train_test_split(X3,Y3, test_size = 0.2, random_state = 5)

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
```

```
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
C = 1
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
C = 10
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
C = 100
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
C = 1000
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
```

In [27]:
```
for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
C = 1
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
C = 10
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
C = 100
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
C = 1000
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
```

In [28]:
```
for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
C = 1
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
C = 10
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
C = 100
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
C = 1000
Training error: 0.019543973941368087
Accuracy: 0.9804560260586319
Cross Validation Score: 0.980439980870397
```

50/50

In [29]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X3,Y3, test_size = 0.5, random_state = 5)

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9739583333333334
C = 1
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9739583333333334
C = 10
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9739583333333334
C = 100
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9713541666666666
C = 1000
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9739583333333334
```

In [30]:
```
for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9739583333333334
C = 1
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9739583333333334
C = 10
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9739583333333334
C = 100
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9713541666666666
C = 1000
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9739583333333334
```

In [31]:
```
for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9739583333333334
C = 1
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9739583333333334
C = 10
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9739583333333334
C = 100
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9713541666666666
C = 1000
Training error: 0.02604166666666663
Accuracy: 0.9739583333333334
Cross Validation Score: 0.9739583333333334
```

20/80

In [32]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X3,Y3, test_size = 0.8, random_state = 5)

for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.02614379084967322
Accuracy: 0.9738562091503268
Cross Validation Score: 0.9738562091503268
C = 1
Training error: 0.02614379084967322
Accuracy: 0.9738562091503268
Cross Validation Score: 0.9738562091503268
C = 10
Training error: 0.02614379084967322
Accuracy: 0.9738562091503268
Cross Validation Score: 0.9673202614379085
C = 100
Training error: 0.0326797385620915
Accuracy: 0.9673202614379085
Cross Validation Score: 0.9673202614379085
C = 1000
Training error: 0.0326797385620915
Accuracy: 0.9673202614379085
Cross Validation Score: 0.9673202614379085
```

In [33]:

```
for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.02614379084967322
Accuracy: 0.9738562091503268
Cross Validation Score: 0.9738562091503268
C = 1
Training error: 0.02614379084967322
Accuracy: 0.9738562091503268
Cross Validation Score: 0.9738562091503268
C = 10
Training error: 0.02614379084967322
Accuracy: 0.9738562091503268
Cross Validation Score: 0.9673202614379085
C = 100
Training error: 0.0326797385620915
Accuracy: 0.9673202614379085
Cross Validation Score: 0.9673202614379085
C = 1000
Training error: 0.0326797385620915
Accuracy: 0.9673202614379085
Cross Validation Score: 0.9673202614379085
```

In [34]:

```
for C in Cs:
    classifier = LogisticRegression(solver = 'liblinear', C = C, class_weight = 'balanced')
    classifier.fit(X_train, Y_train)
    e_training = calc_error(X_train, Y_train, classifier)
    print('C = {}'.format(C))
    print('Training error: {}'.format(e_training[0]))
    print('Accuracy: {}'.format(e_training[1]))
    print('Cross Validation Score: {}'.format(cross_val_score(classifier,X_train,Y_train, cv=3).mean()))
```

```
C = 0.1
Training error: 0.02614379084967322
Accuracy: 0.9738562091503268
Cross Validation Score: 0.9738562091503268
C = 1
Training error: 0.02614379084967322
Accuracy: 0.9738562091503268
Cross Validation Score: 0.9738562091503268
C = 10
Training error: 0.02614379084967322
Accuracy: 0.9738562091503268
Cross Validation Score: 0.9673202614379085
C = 100
Training error: 0.0326797385620915
Accuracy: 0.9673202614379085
Cross Validation Score: 0.9673202614379085
C = 1000
Training error: 0.0326797385620915
Accuracy: 0.9673202614379085
Cross Validation Score: 0.9673202614379085
```