UE208066 Naveen Kumar Meena

In [2]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
review = pd.read_csv("restaurant22.csv")
review
```

Out[2]:

	Reviews	keyword1	frequency1	keyword2	frequency2	Score	class
0	1	delicious	3	terrible	1	6.0	1
1	2	delicious	4	terrible	1	8.5	1
2	3	delicious	2	terrible	1	3.5	1
3	4	delicious	1	terrible	2	-0.5	0
4	5	delicious	1	terrible	2	-0.5	0
5	6	delicious	2	terrible	0	3.5	1
6	7	delicious	1	terrible	1	2.0	1
7	8	delicious	2	terrible	2	2.0	1
8	9	delicious	1	terrible	2	-1.0	0
9	10	delicious	2	terrible	4	-2.0	0

In [3]:

```
print("keyword1 :  ",review['keyword1'].unique())
```

```
keyword1 :   ['delicious']
```

In [4]:

```
print("keyword2 :  ",review['keyword2'].unique())
```

```
keyword2 :   ['terrible']
```

In [5]:

```
review.groupby('keyword1').size()
```

Out[5]:

```
keyword1
delicious    10
dtype: int64
```

In [7]:

```
a = review[['frequency1','frequency2']]
b = np.array(a)
c = review[['class']]
d = np.array(c)
print('b: \n',b)
print('d:\n',d)
```

```
b: 
 [[3 1]
 [4 1]
 [2 1]
 [1 2]
 [1 2]
 [2 0]
 [1 1]
 [2 2]
 [1 2]
 [2 4]]
d:
 [[1]
 [1]
 [1]
 [0]
 [0]
 [1]
 [1]
 [1]
 [0]
 [0]]
```

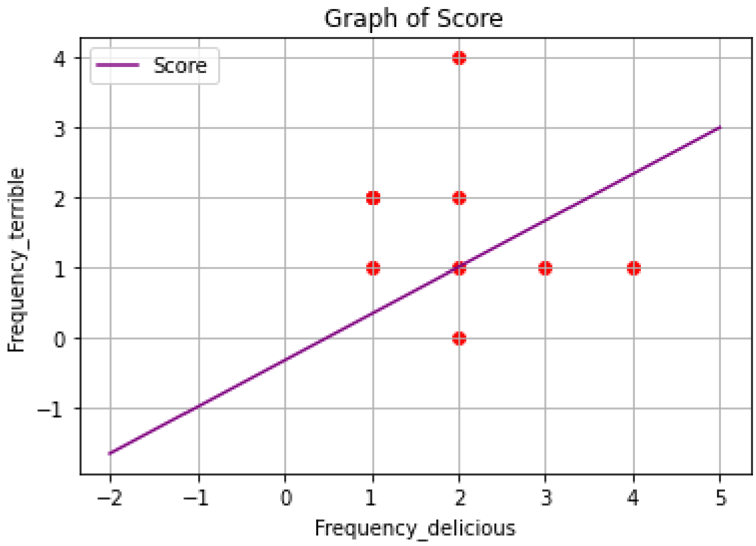
In [40]:

```
# Assignment-1
theta0 = 0
theta1 = 1
theta2 = -1.5
score_review1 = theta0 + (theta1 * b[0][0]) + (theta1 * b[0][1])
print('Score_review1:',score_review1)

scores=[]
for i in range(len(b)):
    scores1 = theta0 + (theta1*b[i][0]) + (theta2*b[i][1])
    scores.append(scores1)
    i+=1
print('\nScores:',scores)

sigmoid_score = 1 / (1 + np.exp(-score_review1))
print('\nSigmoid Score:\n',sigmoid_score)

sig_scores=[]
Yhat=[]
for i in range(len(scores)):
    scores11 = 1 / (1 + np.exp(-scores[i]))
    sig_scores.append(scores11)
    if (scores11>0.5):
        Yhat1 = 1
    else:
        Yhat1 = 0
    Yhat.append(Yhat1)
    i+=1
print('\nSigmoid Scores: \n',sig_scores)
print('\n Yhat: \n', Yhat)

TP = TN = FP = FN = 0
for i in range(len(d)):
    if(d[i] == 1 and Yhat[i] == 1):
        TP = TP + 1
    if(d[i] == 1 and Yhat[i] == 0):
        FN = FN + 1
    if(d[i] == 0 and Yhat[i] == 0):
        TN = TN + 1
    if(d[i] == 0 and Yhat[i] == 1):
        FP = FP + 1

precision = TP / (TP + FN)
recall = TP / (TP+FP)
f1 = (2*precision*recall) / (precision+recall)
accu = (TP+TN)/(TP+TN+FP+FN)

print(precision)
print(recall)
print(f1)
print(accu)
```

```
Score_review1: 4

Scores: [1.5, 2.5, 0.5, -2.0, -2.0, 2.0, -0.5, -1.0, -2.0, -4.0]

Sigmoid Score:
 0.9820137900379085

Sigmoid Scores:
 [0.8175744761936437, 0.9241418199787566, 0.6224593312018546, 0.1192029220
2211755, 0.11920292202211755, 0.8807970779778823, 0.3775406687981454, 0.26
89414213699951, 0.11920292202211755, 0.01798620996209156]

 Yhat:
 [1, 1, 1, 0, 0, 1, 0, 0, 0, 0]
0.6666666666666666
1.0
0.8
0.8
```

In [28]:

```
x = np.array(review['frequency1'])
y = np.array(review['frequency2'])
plt.scatter(x,y,color='red')
X = np.linspace(-2,5,100)
score_review1=0.5
y = (score_review1 - theta0 - (theta1 * X))/theta2
plt.plot(X,y,color='purple',label='Score')
plt.title('Graph of Score')
plt.xlabel("Frequency_delicious")
plt.ylabel("Frequency_terrible")
plt.legend(loc='upper left')
plt.grid()
plt.show()
```



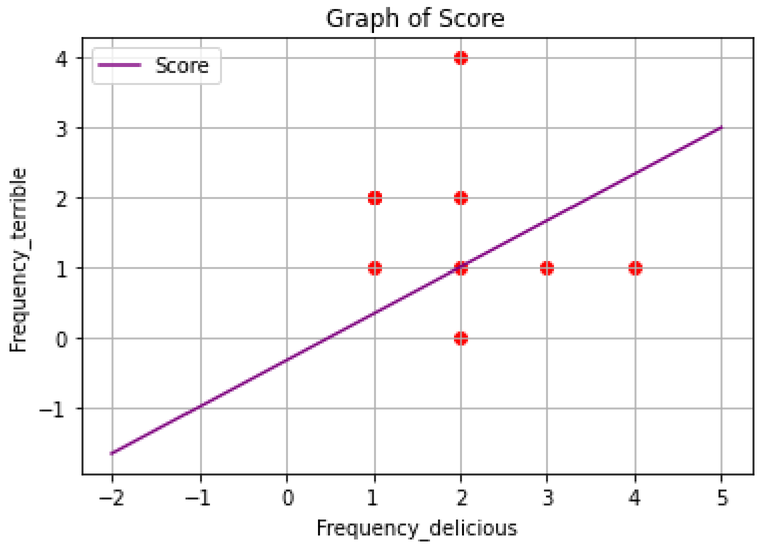
In [39]:

```
# Assignment-2
theta0 = 2
theta1 = 1
theta2 = -1.5
score_review1 = theta0 + (theta1 * b[0][0]) + (theta1 * b[0][1])
print('Score_review1:',score_review1)

scores=[]
for i in range(len(b)):
    scores1 = theta0 + (theta1*b[i][0]) + (theta2*b[i][1])
    scores.append(scores1)
    i+=1
print('\nScores:',scores)

sigmoid_score = 1 / (1 + np.exp(-score_review1))
print('\nSigmoid Score:\n',sigmoid_score)

# print('\nSigmoid Scores:\n',sig_scores)

sig_scores=[]
Yhat=[]
for i in range(len(scores)):
    scores11 = 1 / (1 + np.exp(-scores[i]))
    sig_scores.append(scores11)
    if (scores11>0.5):
        Yhat1 = 1
    else:
        Yhat1 = 0
    Yhat.append(Yhat1)
    i+=1
print('\nSigmoid Scores: \n',sig_scores)
print('\n Yhat: \n', Yhat)

TP = TN = FP = FN = 0
for i in range(len(d)):
    if(d[i] == 1 and Yhat[i] == 1):
        TP = TP + 1
    if(d[i] == 1 and Yhat[i] == 0):
        FN = FN + 1
    if(d[i] == 0 and Yhat[i] == 0):
        TN = TN + 1
    if(d[i] == 0 and Yhat[i] == 1):
        FP = FP + 1

precision = TP / (TP + FN)
recall = TP / (TP+FP)
f1 = (2*precision*recall) / (precision+recall)
accu = (TP+TN)/(TP+TN+FP+FN)

print(precision)
print(recall)
print(f1)
print(accu)
```

```
Score_review1: 6

Scores: [3.5, 4.5, 2.5, 0.0, 0.0, 4.0, 1.5, 1.0, 0.0, -2.0]

Sigmoid Score:
 0.9975273768433653

Sigmoid Scores:
 [0.9706877692486436, 0.9890130573694068, 0.9241418199787566, 0.5, 0.5, 0.
9820137900379085, 0.8175744761936437, 0.7310585786300049, 0.5, 0.119202922
02211755]

 Yhat:
 [1, 1, 1, 0, 0, 1, 1, 1, 0, 0]
1.0
1.0
1.0
1.0
```

In [29]:

```
x = np.array(review['frequency1'])
y = np.array(review['frequency2'])
plt.scatter(x,y,color='red')
X = np.linspace(-2,5,100)
score_review1=0.5
y = (score_review1 - theta0 - (theta1 * X))/theta2
plt.plot(X,y,color='purple',label='Score')
plt.title('Graph of Score')
plt.xlabel("Frequency_delicious")
plt.ylabel("Frequency_terrible")
plt.legend(loc='upper left')
plt.grid()
plt.show()
```



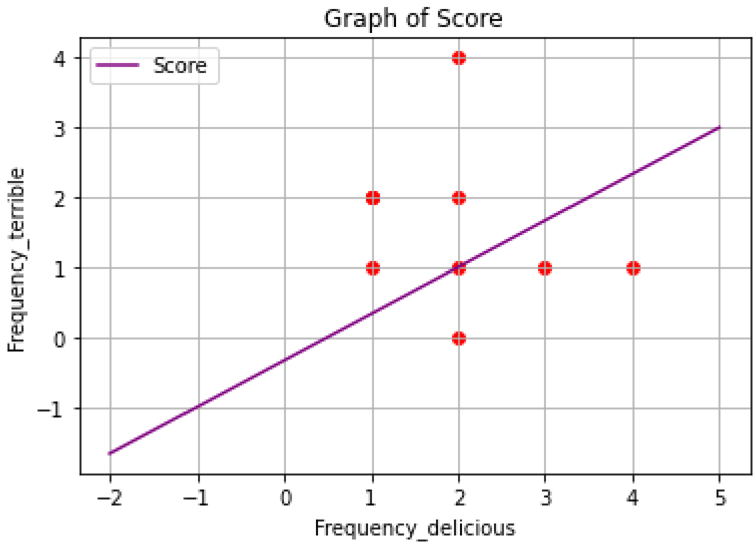
In [38]:

```
# Assignment-3
theta0 = -2
theta1 = 1
theta2 = -1.5
score_review1 = theta0 + (theta1 * b[0][0]) + (theta1 * b[0][1])
print('Score_review1:',score_review1)

scores=[]
for i in range(len(b)):
    scores1 = theta0 + (theta1*b[i][0]) + (theta2*b[i][1])
    scores.append(scores1)
    i+=1
print('\nScores:',scores)

sigmoid_score = 1 / (1 + np.exp(-score_review1))
print('\nSigmoid Score:\n',sigmoid_score)


sig_scores=[]
Yhat=[]
for i in range(len(scores)):
    scores11 = 1 / (1 + np.exp(-scores[i]))
    sig_scores.append(scores11)
    if (scores11>0.5):
        Yhat1 = 1
    else:
        Yhat1 = 0
    Yhat.append(Yhat1)
    i+=1
print('\nSigmoid Scores: \n',sig_scores)
print('\n Yhat: \n', Yhat)

TP = TN = FP = FN = 0
for i in range(len(d)):
    if(d[i] == 1 and Yhat[i] == 1):
        TP = TP + 1
    if(d[i] == 1 and Yhat[i] == 0):
        FN = FN + 1
    if(d[i] == 0 and Yhat[i] == 0):
        TN = TN + 1
    if(d[i] == 0 and Yhat[i] == 1):
        FP = FP + 1

precision = TP / (TP + FN)
recall = TP / (TP+FP)
f1 = (2*precision*recall) / (precision+recall)
accu = (TP+TN)/(TP+TN+FP+FN)

print(precision)
print(recall)
print(f1)
print(accu)
```

```
Score_review1: 2

Scores: [-0.5, 0.5, -1.5, -4.0, -4.0, 0.0, -2.5, -3.0, -4.0, -6.0]

Sigmoid Score:
 0.8807970779778823

Sigmoid Scores:
 [0.3775406687981454, 0.6224593312018546, 0.18242552380635635, 0.017986209
96209156, 0.01798620996209156, 0.5, 0.07585818002124355, 0.047425873177566
78, 0.01798620996209156, 0.0024726231566347743]

 Yhat:
 [0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
0.16666666666666666
1.0
0.2857142857142857
0.5
```

In [30]:

```
x = np.array(review['frequency1'])
y = np.array(review['frequency2'])
plt.scatter(x,y,color='red')
X = np.linspace(-2,5,100)
score_review1=0.5
y = (score_review1 - theta0 - (theta1 * X))/theta2
plt.plot(X,y,color='purple',label='Score')
plt.title('Graph of Score')
plt.xlabel("Frequency_delicious")
plt.ylabel("Frequency_terrible")
plt.legend(loc='upper left')
plt.grid()
plt.show()
```



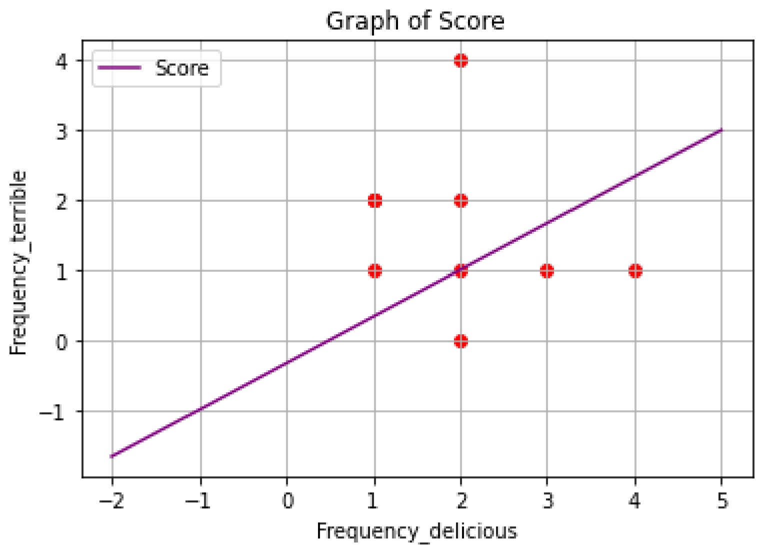
In [37]:

```
# Assignment-4
theta0 = 0
theta1 = 3
theta2 = -3
score_review1 = theta0 + (theta1 * b[0][0]) + (theta1 * b[0][1])
print('Score_review1:',score_review1)

scores=[]
for i in range(len(b)):
    scores1 = theta0 + (theta1*b[i][0]) + (theta2*b[i][1])
    scores.append(scores1)
    i+=1
print('\nScores:',scores)

sigmoid_score = 1 / (1 + np.exp(-score_review1))
# print('\nSigmoid Score:\n',sigmoid_score)

sig_scores=[]
Yhat=[]
for i in range(len(scores)):
    scores11 = 1 / (1 + np.exp(-scores[i]))
    sig_scores.append(scores11)
    if (scores11>0.5):
        Yhat1 = 1
    else:
        Yhat1 = 0
    Yhat.append(Yhat1)
    i+=1
print('\nSigmoid Scores: \n',sig_scores)
print('\n Yhat: \n', Yhat)

TP = TN = FP = FN = 0
for i in range(len(d)):
    if(d[i] == 1 and Yhat[i] == 1):
        TP = TP + 1
    if(d[i] == 1 and Yhat[i] == 0):
        FN = FN + 1
    if(d[i] == 0 and Yhat[i] == 0):
        TN = TN + 1
    if(d[i] == 0 and Yhat[i] == 1):
        FP = FP + 1

precision = TP / (TP + FN)
recall = TP / (TP+FP)
f1 = (2*precision*recall) / (precision+recall)
accu = (TP+TN)/(TP+TN+FP+FN)

print(precision)
print(recall)
print(f1)
print(accu)
```

```
Score_review1: 12

Scores: [6, 9, 3, -3, -3, 6, 0, 0, -3, -6]

Sigmoid Scores:
 [0.9975273768433653, 0.9998766054240137, 0.9525741268224334, 0.0474258731
7756678, 0.04742587317756678, 0.9975273768433653, 0.5, 0.5, 0.047425873177
56678, 0.0024726231566347743]

 Yhat:
 [1, 1, 1, 0, 0, 1, 0, 0, 0, 0]
0.6666666666666666
1.0
0.8
0.8
```

In [31]:

```
x = np.array(review['frequency1'])
y = np.array(review['frequency2'])
plt.scatter(x,y,color='red')
X = np.linspace(-2,5,100)
score_review1=0.5
y = (score_review1 - theta0 - (theta1 * X))/theta2
plt.plot(X,y,color='purple',label='Score')
plt.title('Graph of Score')
plt.xlabel("Frequency_delicious")
plt.ylabel("Frequency_terrible")
plt.legend(loc='upper left')
plt.grid()
plt.show()
```



Fruit Data

In [44]:

```
%matplotlib inline
fruits = pd.read_table('fruit_data_with_colors.txt')
fruits
```

Out[44]:

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.8	0.59
2	1	apple	granny_smith	176	7.4	7.2	0.60
3	2	mandarin	mandarin	86	6.2	4.7	0.80
4	2	mandarin	mandarin	84	6.0	4.6	0.79
5	2	mandarin	mandarin	80	5.8	4.3	0.77
6	2	mandarin	mandarin	80	5.9	4.3	0.81
7	2	mandarin	mandarin	76	5.8	4.0	0.81
8	1	apple	braeburn	178	7.1	7.8	0.92
9	1	apple	braeburn	172	7.4	7.0	0.89
10	1	apple	braeburn	166	6.9	7.3	0.93
11	1	apple	braeburn	172	7.1	7.6	0.92
12	1	apple	braeburn	154	7.0	7.1	0.88
13	1	apple	golden_delicious	164	7.3	7.7	0.70
14	1	apple	golden_delicious	152	7.6	7.3	0.69
15	1	apple	golden_delicious	156	7.7	7.1	0.69
16	1	apple	golden_delicious	156	7.6	7.5	0.67
17	1	apple	golden_delicious	168	7.5	7.6	0.73
18	1	apple	cripps_pink	162	7.5	7.1	0.83
19	1	apple	cripps_pink	162	7.4	7.2	0.85
20	1	apple	cripps_pink	160	7.5	7.5	0.86
21	1	apple	cripps_pink	156	7.4	7.4	0.84
22	1	apple	cripps_pink	140	7.3	7.1	0.87
23	1	apple	cripps_pink	170	7.6	7.9	0.88
24	3	orange	spanish_jumbo	342	9.0	9.4	0.75
25	3	orange	spanish_jumbo	356	9.2	9.2	0.75
26	3	orange	spanish_jumbo	362	9.6	9.2	0.74
27	3	orange	selected_seconds	204	7.5	9.2	0.77
28	3	orange	selected_seconds	140	6.7	7.1	0.72
29	3	orange	selected_seconds	160	7.0	7.4	0.81
30	3	orange	selected_seconds	158	7.1	7.5	0.79
31	3	orange	selected_seconds	210	7.8	8.0	0.82
32	3	orange	selected_seconds	164	7.2	7.0	0.80
33	3	orange	turkey_navel	190	7.5	8.1	0.74
34	3	orange	turkey_navel	142	7.6	7.8	0.75
35	3	orange	turkey_navel	150	7.1	7.9	0.75
36	3	orange	turkey_navel	160	7.1	7.6	0.76

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
37	3	orange	turkey_navel	154	7.3	7.3	0.79
38	3	orange	turkey_navel	158	7.2	7.8	0.77
39	3	orange	turkey_navel	144	6.8	7.4	0.75
40	3	orange	turkey_navel	154	7.1	7.5	0.78
41	3	orange	turkey_navel	180	7.6	8.2	0.79
42	3	orange	turkey_navel	154	7.2	7.2	0.82
43	4	lemon	spanish_belsan	194	7.2	10.3	0.70
44	4	lemon	spanish_belsan	200	7.3	10.5	0.72
45	4	lemon	spanish_belsan	186	7.2	9.2	0.72
46	4	lemon	spanish_belsan	216	7.3	10.2	0.71
47	4	lemon	spanish_belsan	196	7.3	9.7	0.72
48	4	lemon	spanish_belsan	174	7.3	10.1	0.72
49	4	lemon	unknown	132	5.8	8.7	0.73
50	4	lemon	unknown	130	6.0	8.2	0.71
51	4	lemon	unknown	116	6.0	7.5	0.72
52	4	lemon	unknown	118	5.9	8.0	0.72
53	4	lemon	unknown	120	6.0	8.4	0.74
54	4	lemon	unknown	116	6.1	8.5	0.71
55	4	lemon	unknown	116	6.3	7.7	0.72
56	4	lemon	unknown	116	5.9	8.1	0.73
57	4	lemon	unknown	152	6.5	8.5	0.72
58	4	lemon	unknown	118	6.1	8.1	0.70

In [45]:

```
print(fruits.shape)
```

```
(59, 7)
```

In [46]:

```
print(fruits['fruit_name'].unique())
```

```
['apple' 'mandarin' 'orange' 'lemon']
```

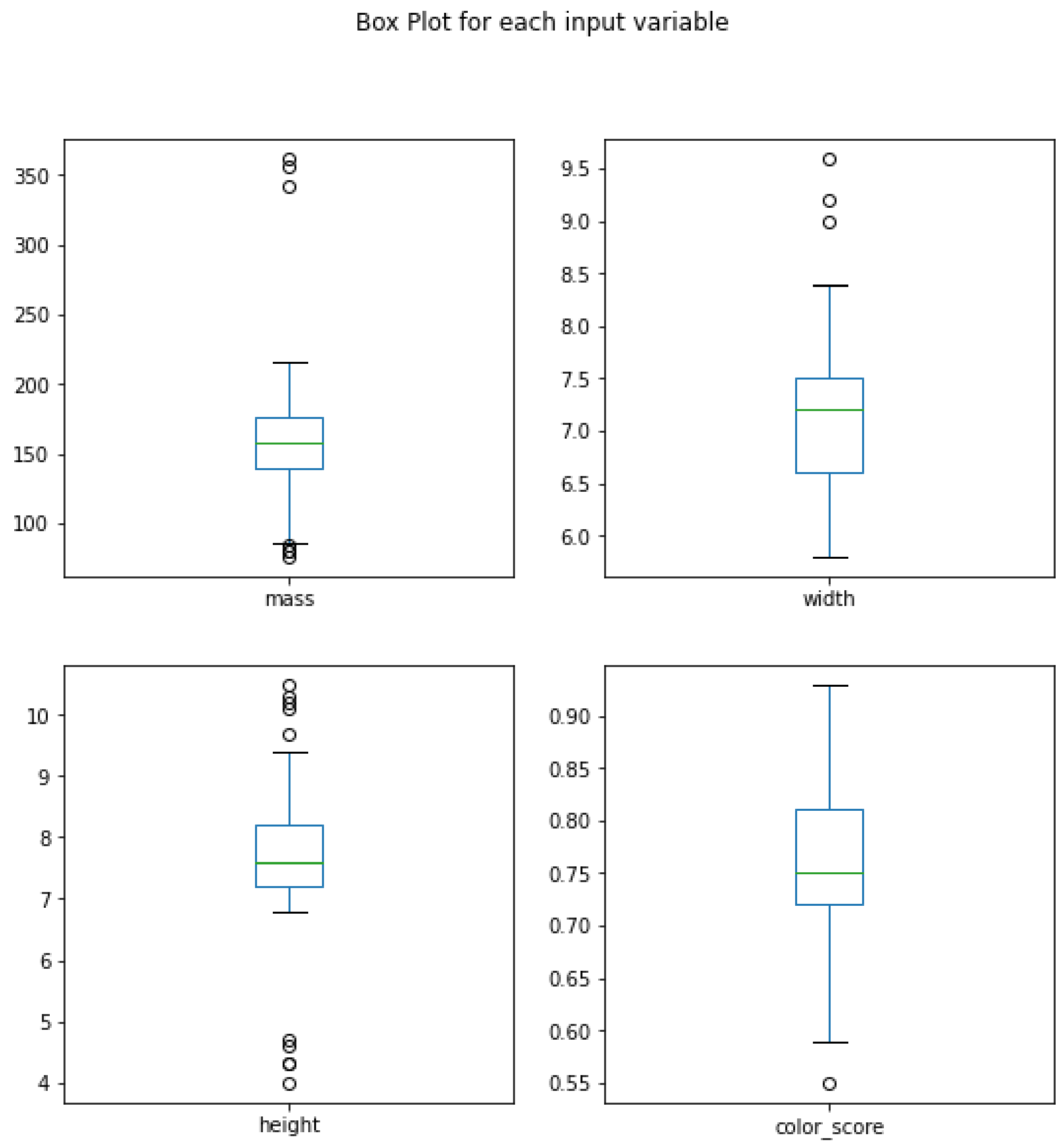
In [47]:

```
print(fruits.groupby('fruit_name').size())
```

```
fruit_name
apple       19
lemon       16
mandarin     5
orange      19
dtype: int64
```

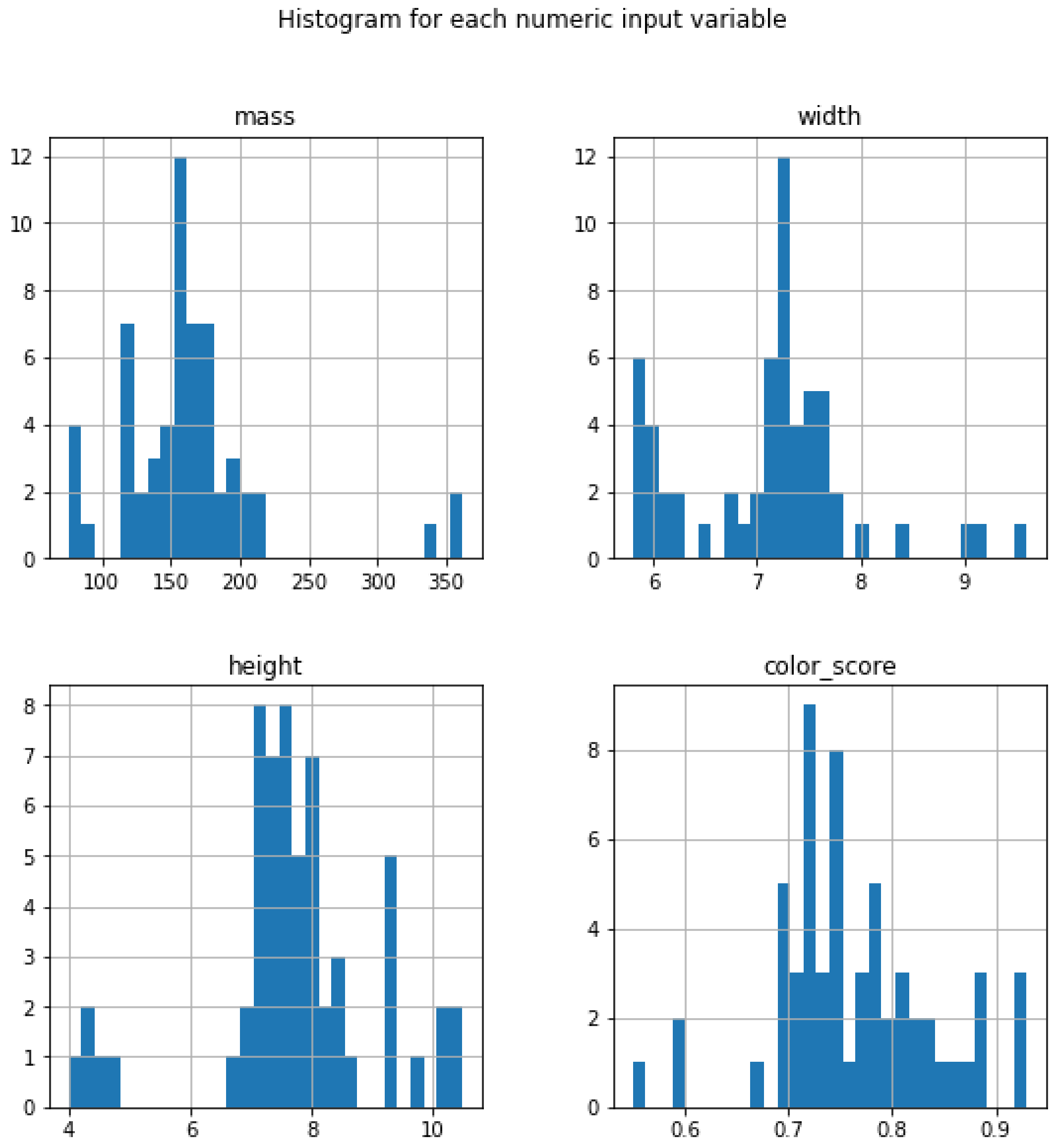
In [48]:

```
fruits.drop('fruit_label', axis=1).plot(kind='box', subplots=True, layout=(2,2), sharex=
title='Box Plot for each input variable')
plt.savefig('fruits_box')
plt.show()
```



In [49]:

```
import pylab as pl
fruits.drop('fruit_label' ,axis=1).hist(bins=30, figsize=(9,9))
pl.suptitle("Histogram for each numeric input variable")
plt.savefig('fruits_hist')
plt.show()
```



In [64]:

```
feature_names = ['mass', 'width', 'height', 'color_score']
X = fruits[feature_names]
y = fruits['fruit_label']
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=0.04 ,random_state=0)
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [65]:

```
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
print('Accuracy of Logistic regression classifier on training set: {:.2f}'.format(logreg
print('Accuracy of Logistic regression classifier on test set: {:.2f}'.format(logreg.sco
```

```
Accuracy of Logistic regression classifier on training set: 0.71
Accuracy of Logistic regression classifier on test set: 0.67
```