In [26]:
```
import pandas as pd
import numpy as np
import sklearn
import matplotlib.pyplot as plt
from sklearn.model_selection \
import train_test_split
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler , LabelEncoder
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
```

Question#1. what is the equation for logistic regression that your classi- fier found from year 1 data?

In [27]:
```
df=pd.read_csv("NVDA_weekly_return_volatility.csv")
year=df['Year'].unique()
Q1_label=[]
yearly_mean=df.groupby('Year')['mean_return'].mean().values
for i in range(len(year)):
    for j in range(len(df)):
        if df['Year'][j]==year[i] and df["mean_return"][j]>yearly_mean[i]:
            Q1_label.append('green')
        elif df['Year'][j]==year[i]:
            Q1_label.append('red')
df['label']=Q1_label
Q1_X=df[df["Year"]==2017][["mean_return","volatility"]]
Q1_y=df[df["Year"]==2017]["label"]
Q1_log_reg_classifier = LogisticRegression()
Q1_log_reg_classifier.fit(Q1_X, Q1_y)
print(Q1_log_reg_classifier.coef_)
print(Q1_log_reg_classifier.intercept_)
print("the logistic regression is 1 / (1+e^-(0.77379469-3.12922686*X1+0.0060517
```

```
[[-3.12922686 -0.0060517 ]]
[0.77379469]
the logistic regression is 1 / (1+e^-(0.77379469-3.12922686*X1+0.0060517*X2))
```

Question#2. what is the accuracy for year 2?

In [28]:
```
Q2_X=df.loc[df["Year"]==2018][["mean_return","volatility"]]
Q2_y=df.loc[df["Year"]==2018]["label"]
print("the year2 accuracy is",accuracy_score(Q2_y, Q1_log_reg_classifier.predic
```

```
the year2 accuracy is 0.8679245283018868
```

Question#3. compute the confusion matrix for year 2

In [29]:
```
a= confusion_matrix(Q2_y, Q1_log_reg_classifier.predict(Q2_X))
print('the confusion matrix is\n',a)
```

```
the confusion matrix is
 [[22  7]
 [ 0 24]]
```

Quetion#4. what is true positive rate (sensitivity or recall) and true negative rate (specificity) for year 2?

In [30]:
```
Q4_TN, Q4_FP, Q4_FN, Q4_TP = confusion_matrix(Q2_y, Q1_log_reg_classifier.predi
Q4_TPR=Q4_TP/(Q4_TP+Q4_FN)
Q4_TNR=Q4_TN/(Q4_TN+Q4_FP)
print('true positive rate',Q4_TPR)
print('true negative rate',Q4_TNR)
```

```
true positive rate 1.0
true negative rate 0.7586206896551724
```