Ishika kesarwani roll no. 92 SAP ID 500076373

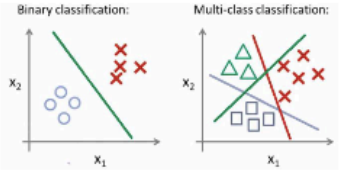
Pattern and Anomaly Detection Lab 4

LOGISTIC REGRESSION

GITHUB LINK- https://github.com/ishikkkkaaaa/UPES/tree/master/Pattern-and-Anomoly-Detection/LAB4%20LOGISTIC%20REGRESSION

```
Create data

Visualize it

Data preparation

Model building (using inbuilt functions)
Training and prediction

Confusion matrix
```

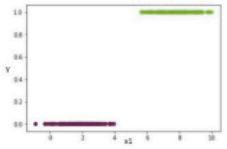
LOGISTIC REGRESSION: Binary classification model



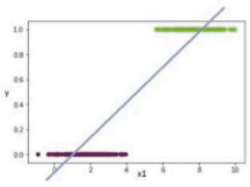
Why the name includes regression and not classification???

It uses the regression inside to be the classification algorithm.

Given X or (Set of x values) we need to predict whether Y is 0 or 1 (Yes/No).

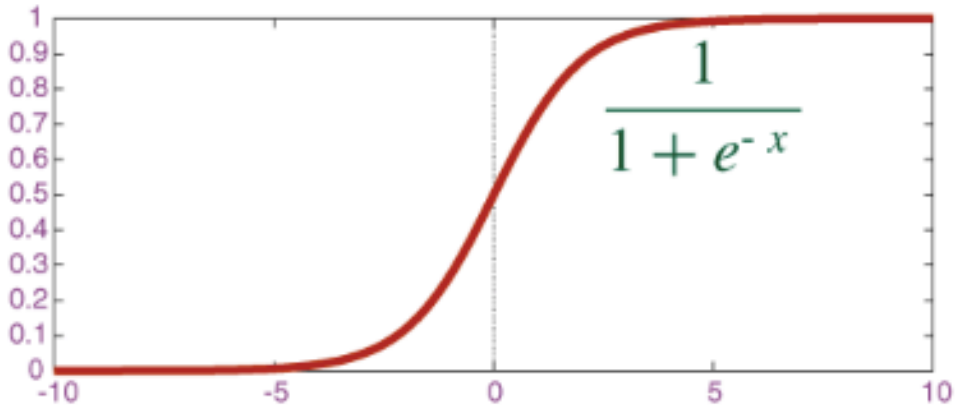


With **Linear Regression (y = mx + c)**



We only accept the values between 0 and 1. How do we manage that?

Solution: Sigmoid function



We first apply the linear equation and apply Sigmoid function for the result so we get the value which is between 0 and 1.

Logistic regression cost function

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \text{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$= -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))]$$

$$P(y=1 \mid x;\theta) = h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

Taken from Prof. Andrew Ng.'s Coursera ML course

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \text{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

Note: $y = 0$ or 1 always

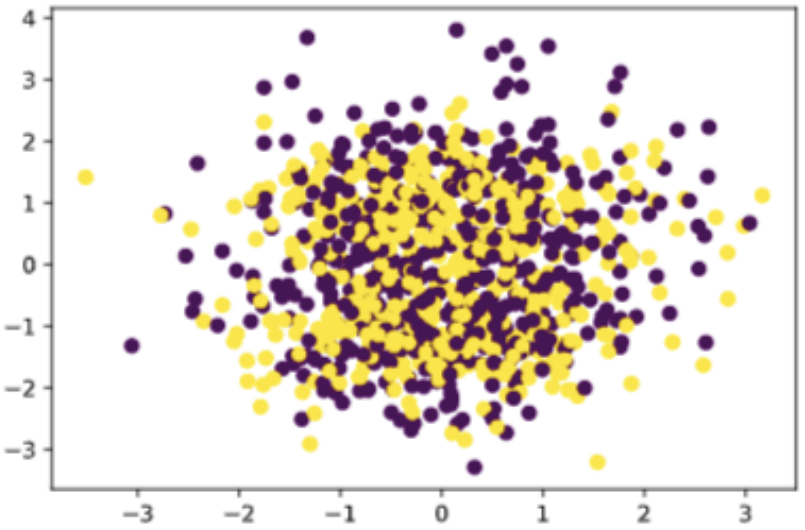
LOGISTIC REGRESSION

In [61]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [62]:
```
#create a dataset using sklearn for binary classification
from sklearn.datasets import make_classification
x, y = make_classification(n_samples=1000, n_features=20,n_classes=2,random_state=30)
```

In [68]:
```
#scatter plot of x and y coordinates
plt.scatter(x[:,0],x[:,1],c=y)
```

Out[68]:
```
<matplotlib.collections.PathCollection at 0x7fba39f3fe80>
```



In [69]:
```
#scatter plot of x and y
print(x.shape)
print(y.shape)
```

```
(1000, 20)
(1000,)
```

In [87]:
```
# split the data into train and test
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=30)
```

In [88]:
```
# logistic regression
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
logreg.fit(x_train, y_train)
```

In [89]:
```
# predict the test data
y_pred = logreg.predict(x_test)
```

In [90]:
```
#validation and evaluation of model
from sklearn.metrics import accuracy_score,confusion_matrix
print(accuracy_score(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
0.92
[[102   6]
 [ 10  82]]
```

In [91]:
```
# classification report
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.91      0.94      0.93       108
           1       0.93      0.89      0.91        92

    accuracy                           0.92       200
   macro avg       0.92      0.92      0.92       200
weighted avg       0.92      0.92      0.92       200
```

In [92]:
```
# confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True)
```

Out[92]: <AxesSubplot:>