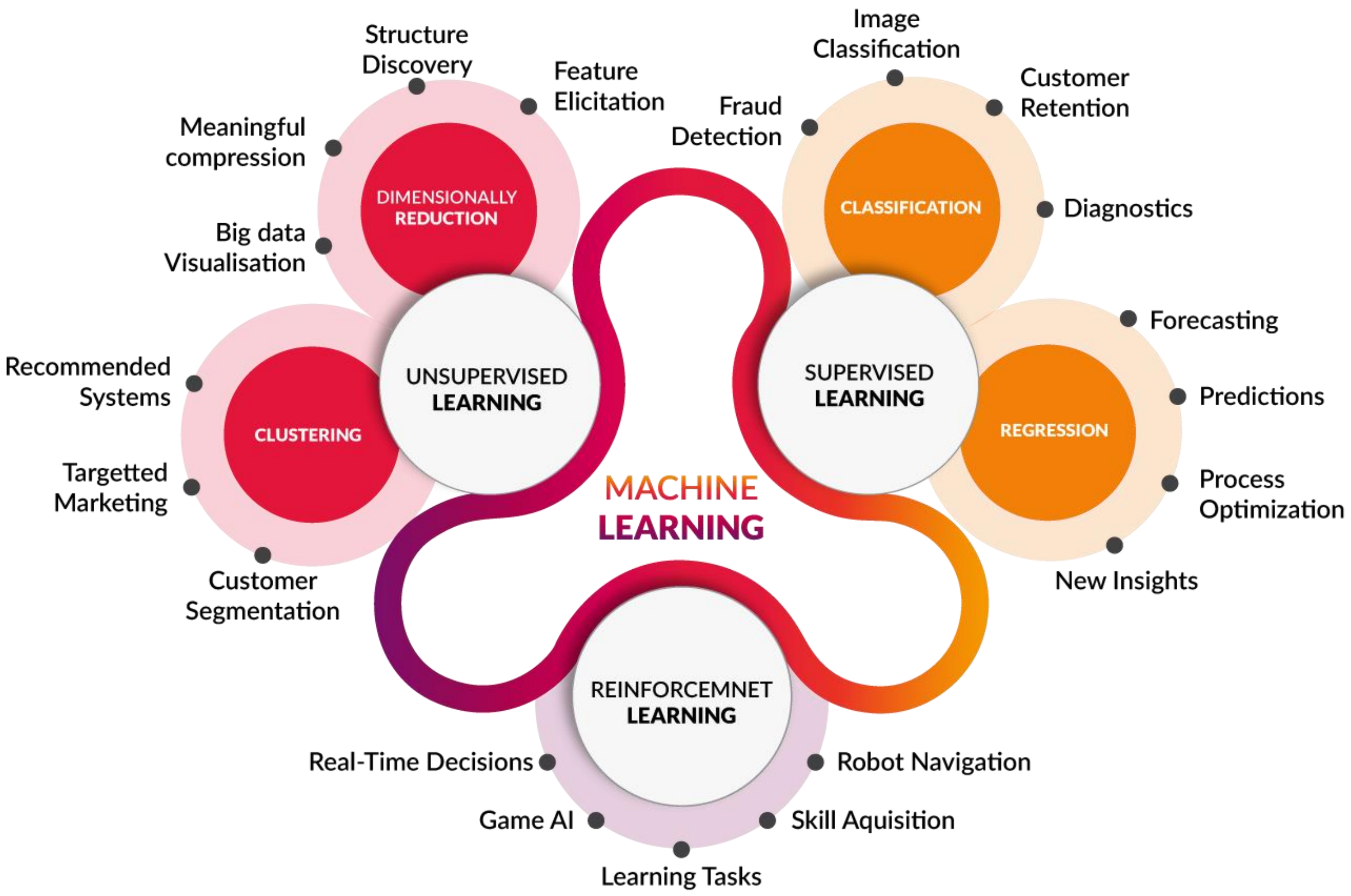
Logistic Regression

Machine Learning

Dr. Adnan Abid

Courtesy Super Data Science



MACHINE LEARNING
UNSUPERVISED LEARNING
DIMENSIONALLY REDUCTION
Structure Discovery
Feature Elicitation
Meaningful compression
Big data Visualisation
CLUSTERING
Recommended Systems
Targetted Marketing
Customer Segmentation
SUPERVISED LEARNING
CLASSIFICATION
Image Classification
Customer Retention
Fraud Detection
Diagnostics
REGRESSION
Forecasting
Predictions
Process Optimization
New Insights
REINFORCEMNET LEARNING
Real-Time Decisions
Robot Navigation
Game AI
Skill Aquisition
Learning Tasks

Logistic Regression

Linear Regression:

- **Simple:**

$$y = b_0 + b_1*x$$

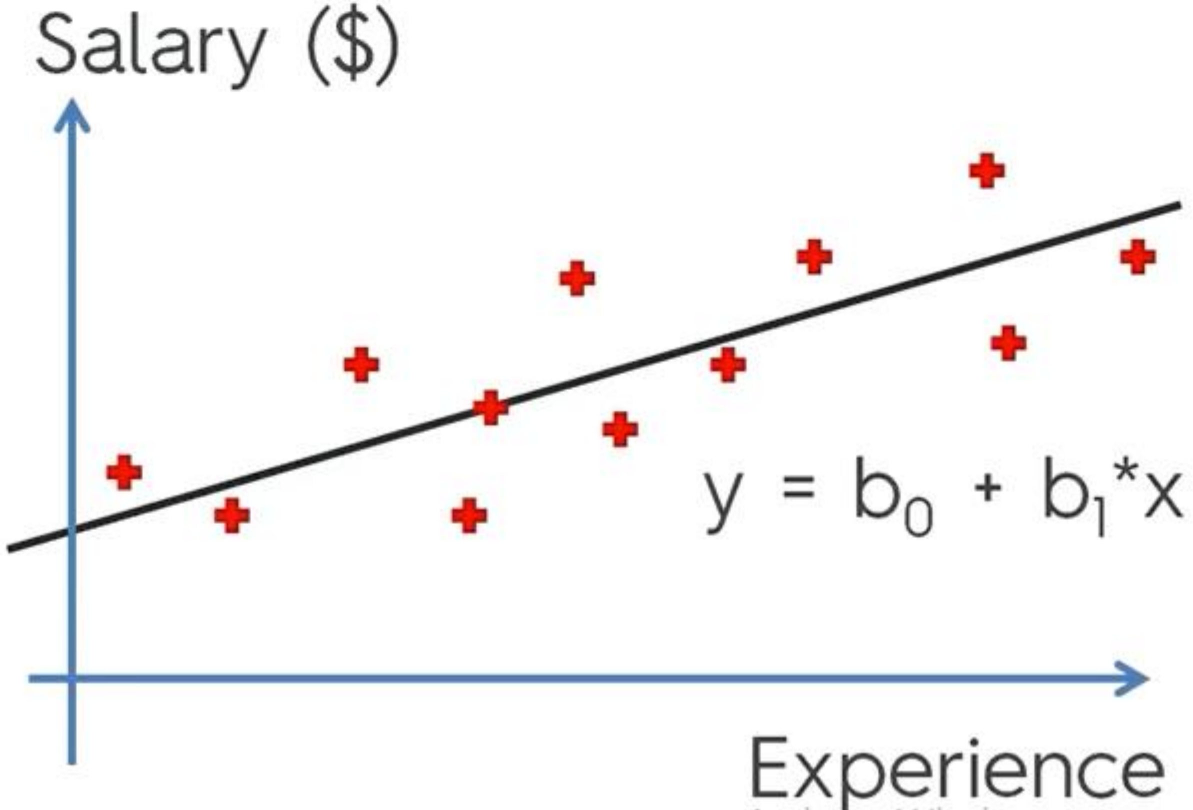
- **Multiple:**

$$y = b_0 + b_1*x_1 + \ldots + b_n*x_n$$

Logistic Regression

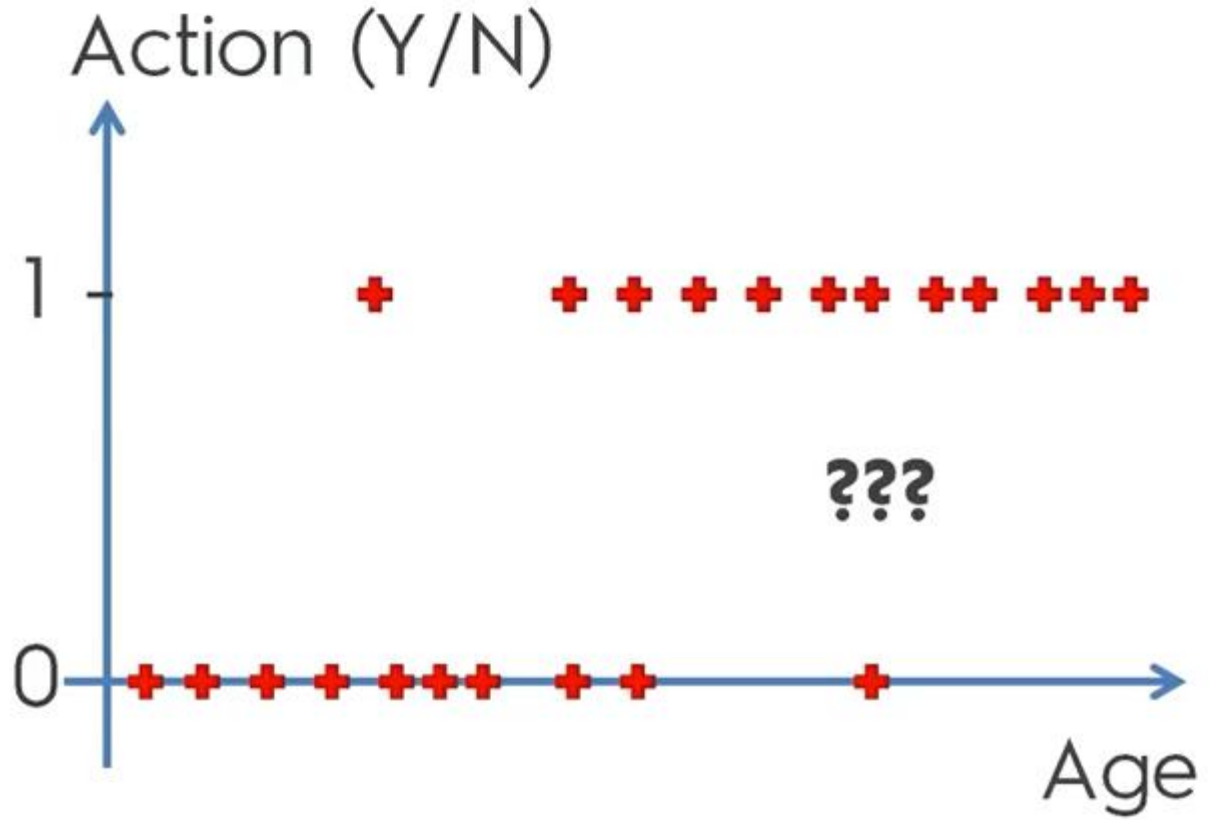
This is new:

We know this:

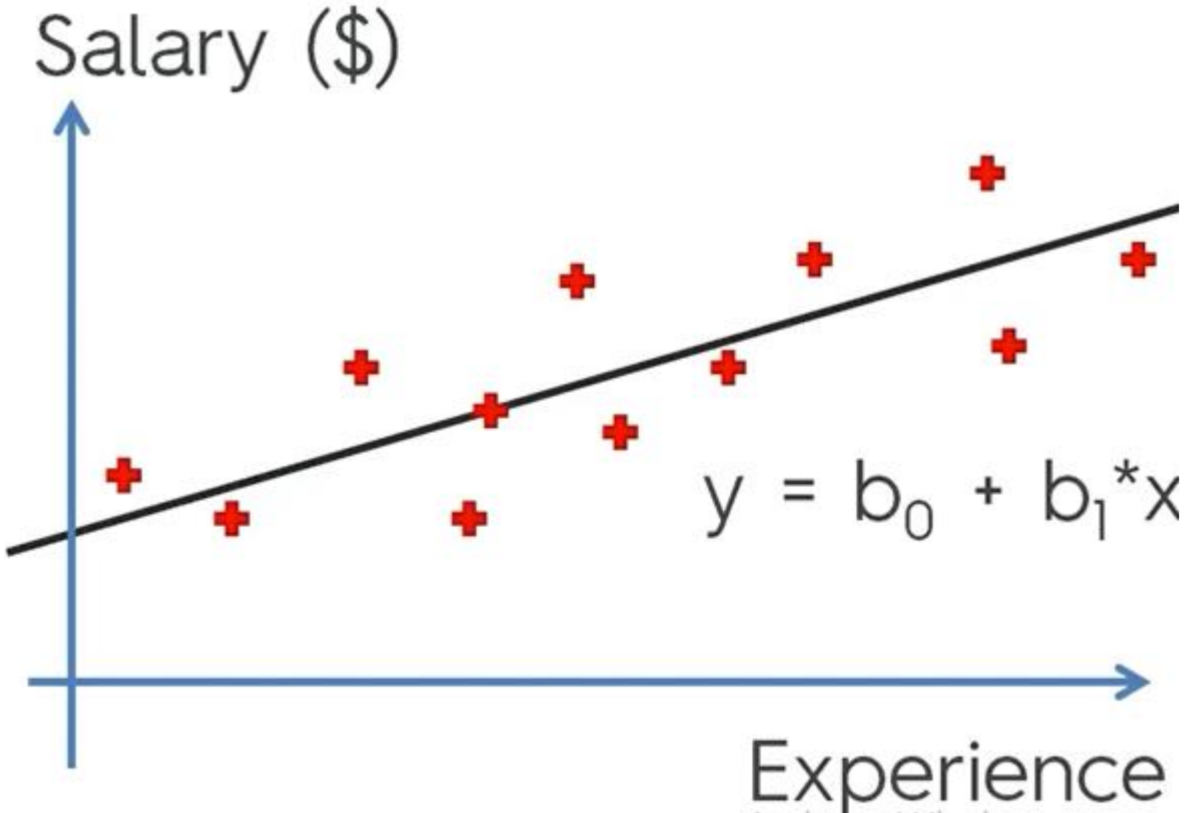


Logistic Regression

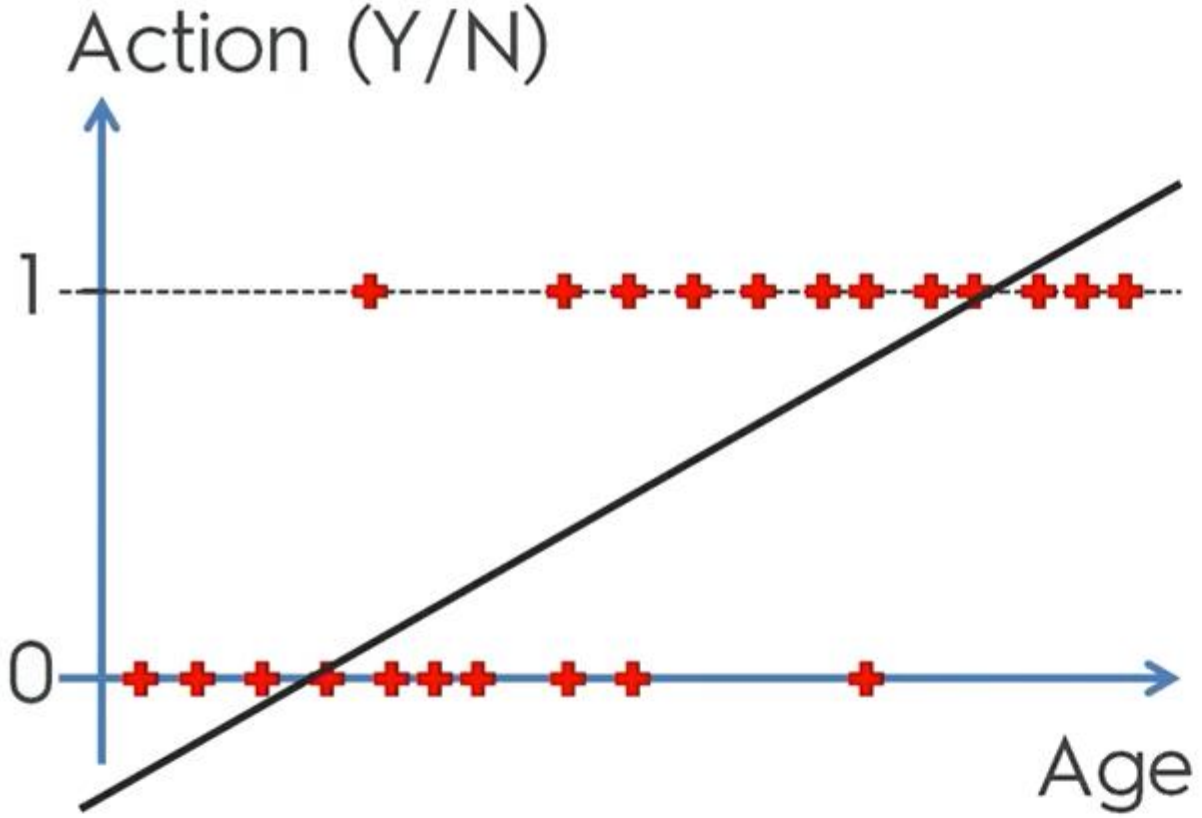
This is new:



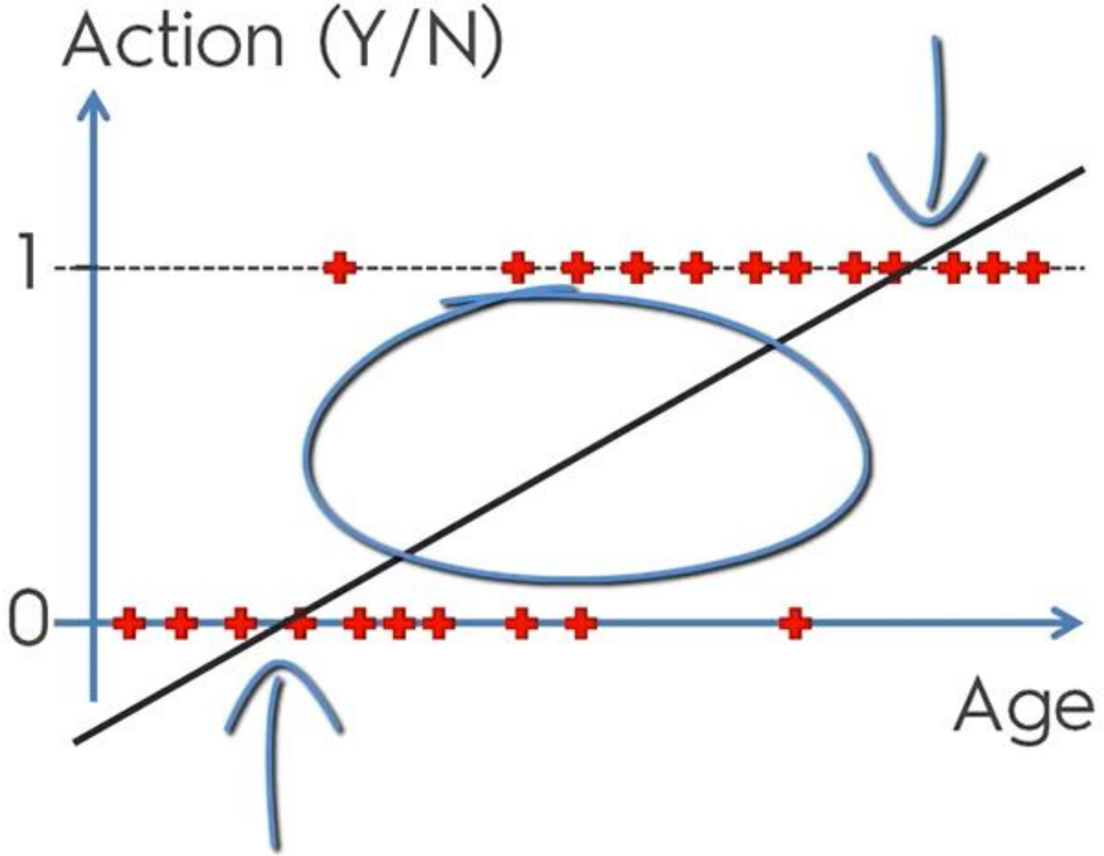
We know this:



Logistic Regression

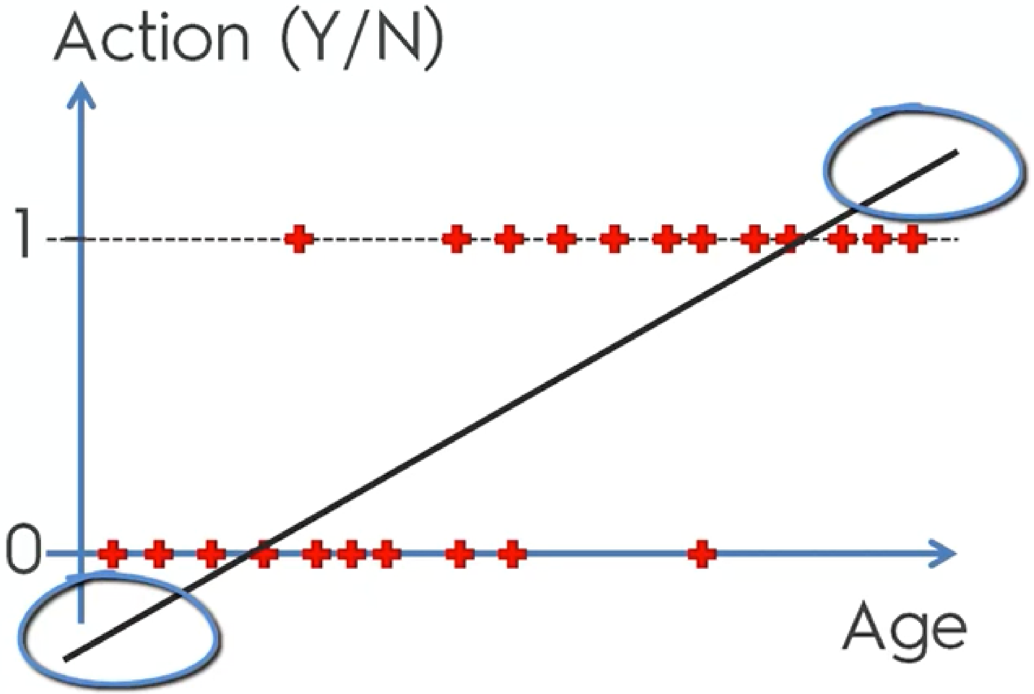


Logistic Regression



The graph shows the probability for the people between the age of 30 and 50 taking up this offer. While as the age increases there is a greater probability for them taking the offer.

Logistic Regression

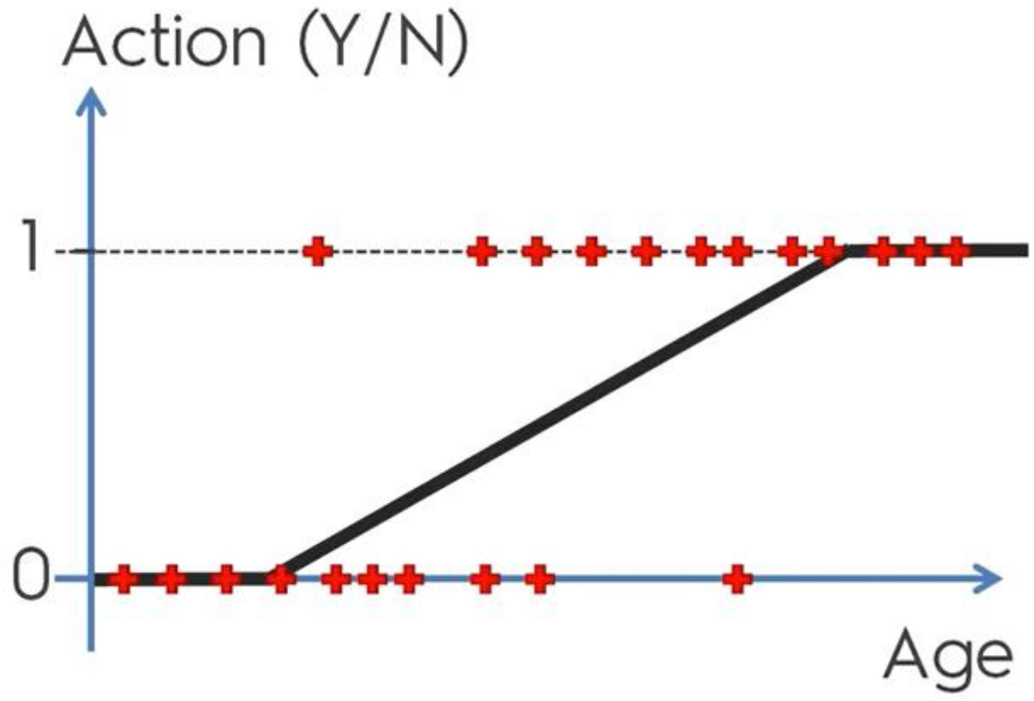


The graph shows the probability for the people between the age of 30 and 50 taking up this offer. While as the age increases there is a greater probability for them taking the offer.

However, for people below 30 the probability seems to be –ve, and for the ones above 50 the probability seems to be greater than 1. Both conditions are not possible.

In simple words the people below 30 are not going to take it, and the ones above 50 will for sure take it.

Logistic Regression



The graph shows the probability for the people between the age of 30 and 50 taking up this offer. While as the age increases there is a greater probability for them taking the offer.

However, for people below 30 the probability seems to be –ve, and for the ones above 50 the probability seems to be greater than 1. Both conditions are not possible.

In simple words the people below 30 are not going to take it, and the ones above 50 will for sure take it.

So we cut those bits off and linear regression looks like this.

Logistic Regression

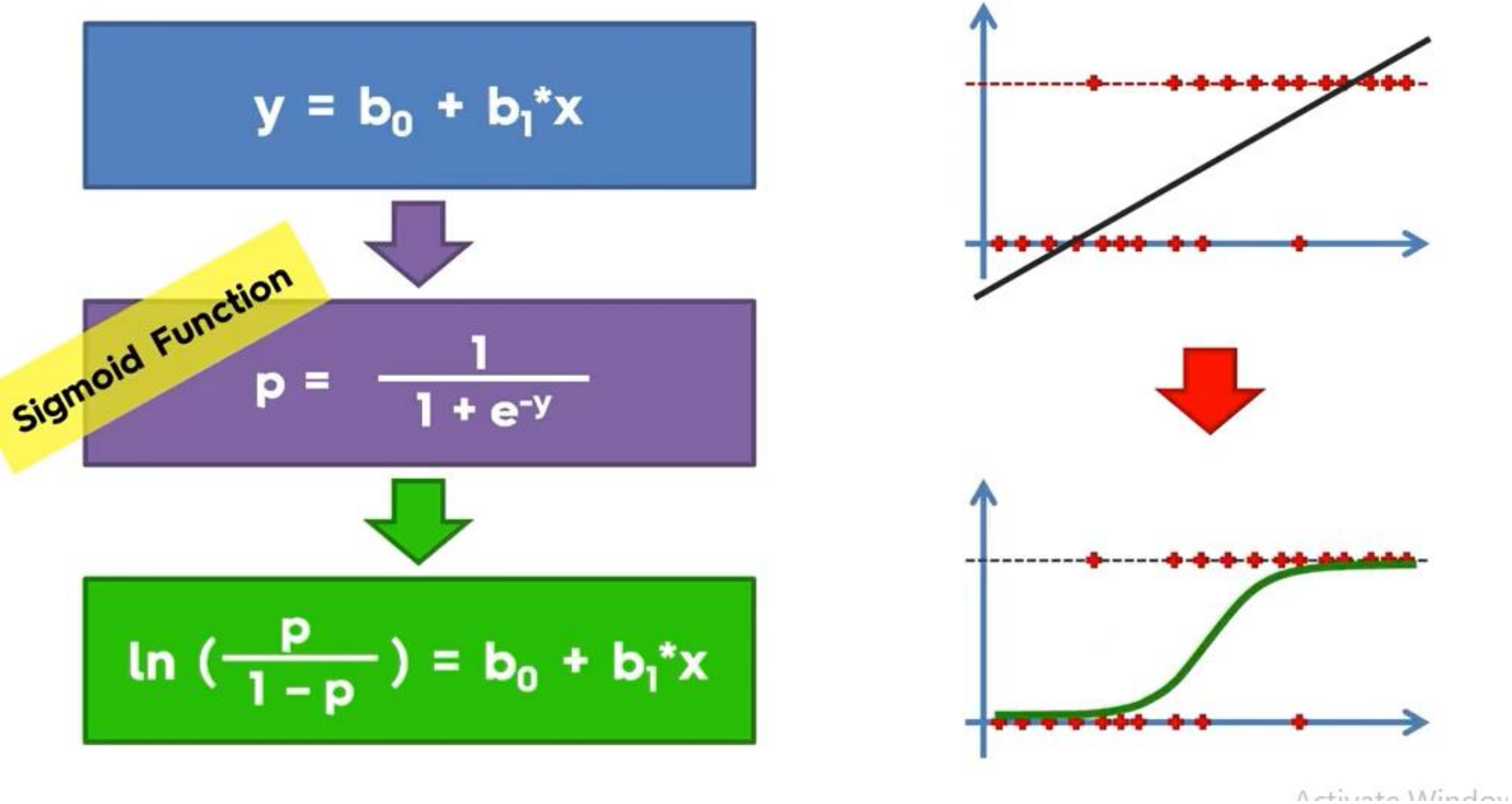
Scientific Explanation for Logistic Regression

____

Blue box is the equation for simple linear regression.

If we apply Sigmoid Function to this equation which is shown in purple box, can calculate y from it.

The we place this y in the blue box and solve it, then we come up with the expression in green box.
So, the equation for linear regression would look like this.
Which is logistic regression function.



$p = 1/(1+e^{-y})$

$p * (1+e^{-y}) = 1$

$1 + e^{-y} = 1/p$

$e^{-y} = 1/p - 1$

$e^{-y} = (1-p)/p$

$e^{y} = p/(1-p)$

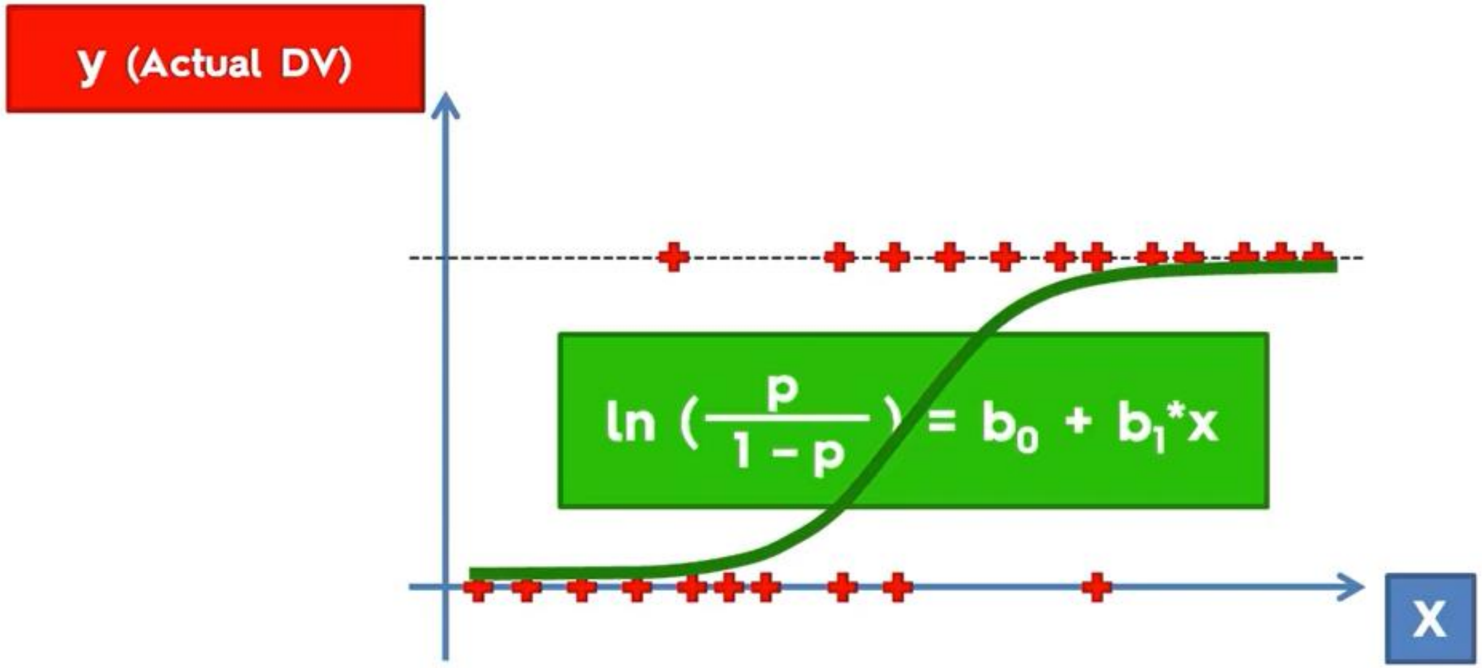
$\ln(e^{y}) = \ln(p/(1-p))$

$y.\ \ln(e) = \ln(p/(1-p))$

$y.1 = \ln(p/(1-p))$

$y = \ln(p/(1-p))$

Logistic Regression



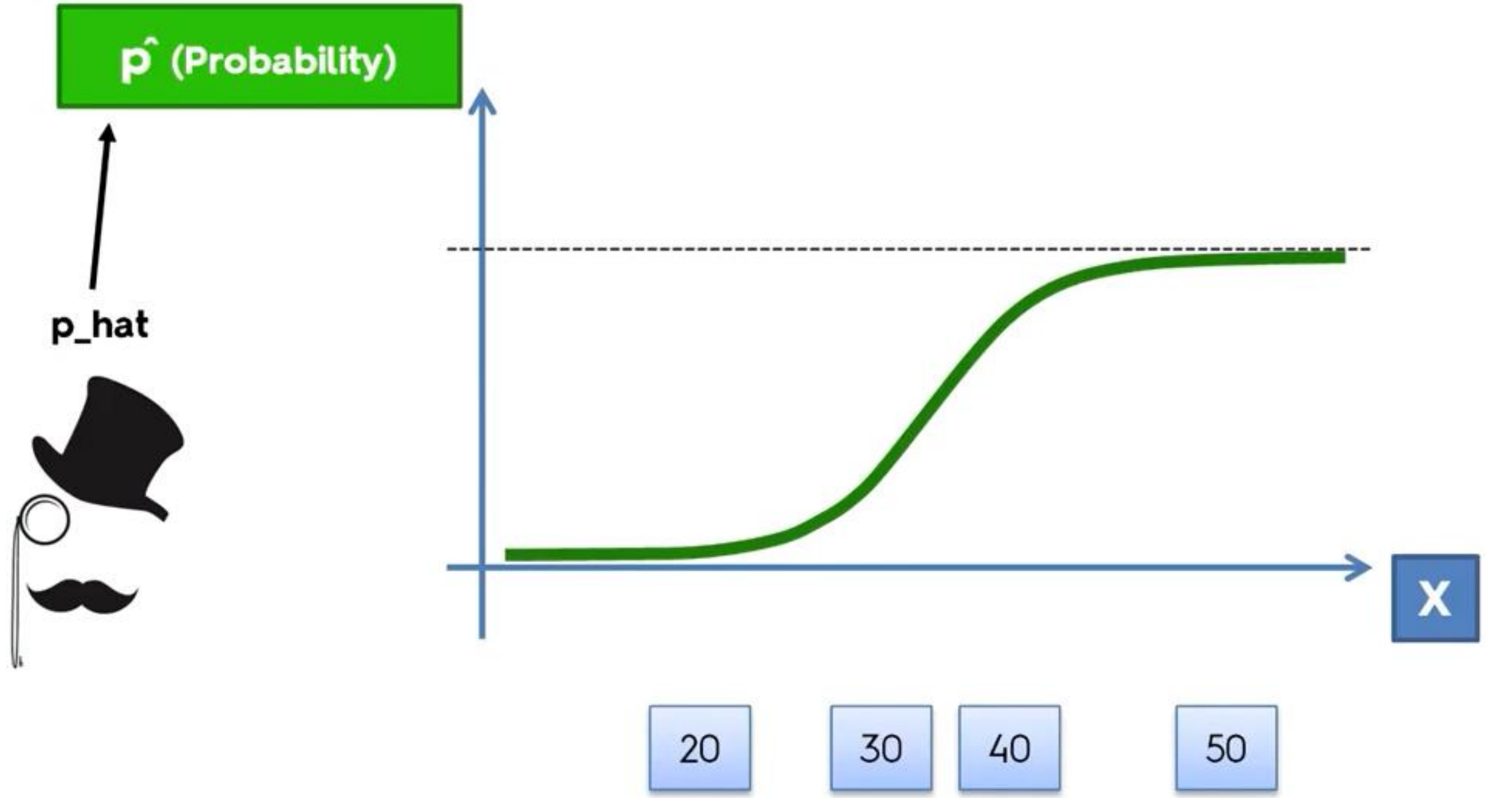
Scientific Explanation for Logistic Regression

____

The graph shows ‘x’ the independent variable, ‘y’ the dependent variable, and a trend-line using the logistic regression formula and the data points.

Again, like simple linear regression we can draw many different lines for the given data and find the best fit line.

Logistic Regression



Scientific Explanation for Logistic Regression

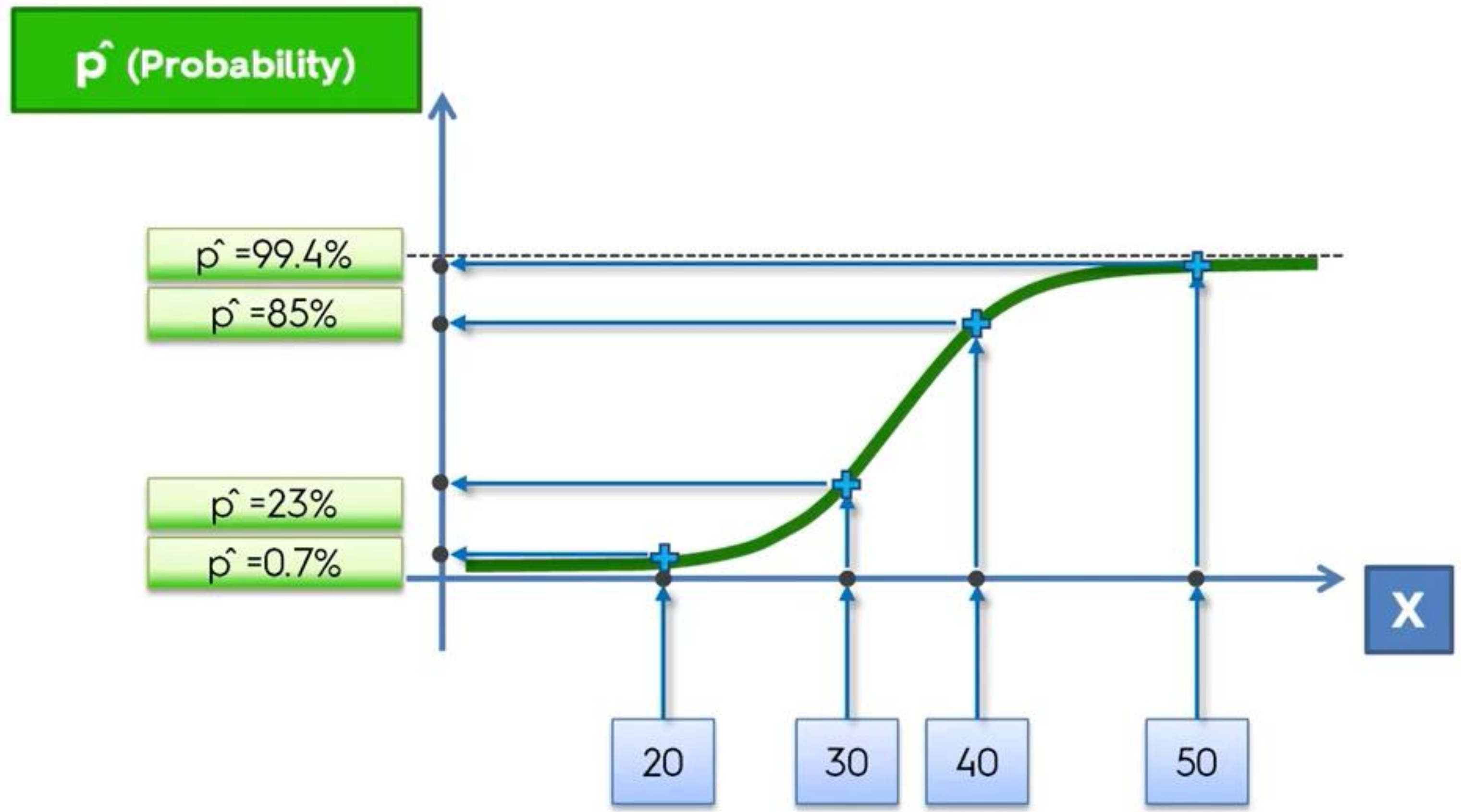
____

The graph shows 'x' the independent variable, 'y' the dependent variable, and a trend-line using the logistic regression formula and the data points.

Again, like simple linear regression we can draw many different lines for the given data and find the best fit line.

We actually calculate probability (p-hat) for the y value for different possible x values; i.e. we don't compute y-actual but we calculate y-hat i.e. predicted y variable.

Logistic Regression



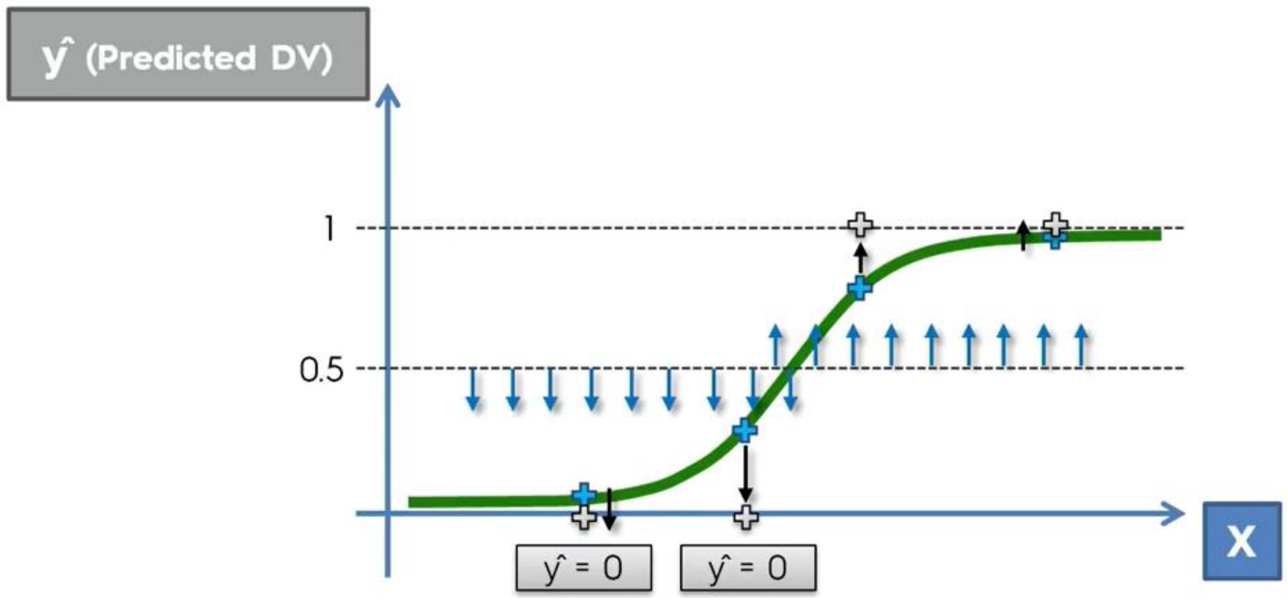
Scientific Explanation for Logistic Regression

____

As, we actually calculate probability (p-hat) for the y value for different possible x values; i.e. we don't compute y-actual but we calculate y-hat i.e. predicted y variable.

Here we can see some p-hat values which show the probability of taking up to the offer for different sample age groups.

Logistic Regression



Scientific Explanation for Logistic Regression

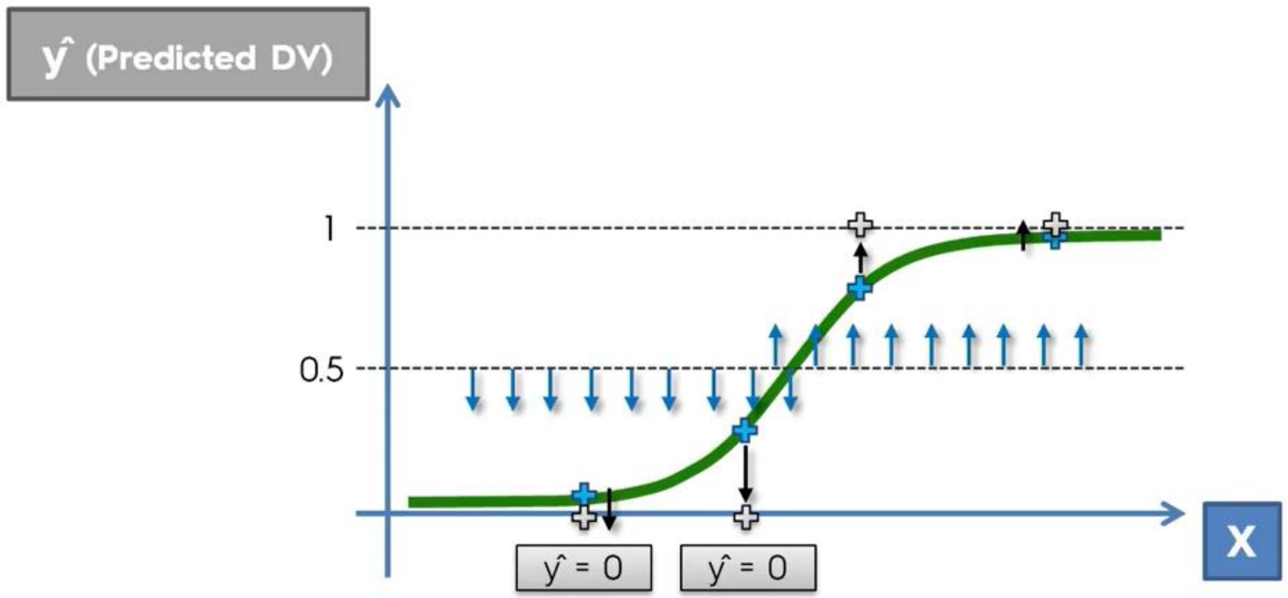
____

As, we actually calculate probability (p-hat) for the y value for different possible x values; i.e. we don't compute y-actual but we calculate y-hat i.e. predicted y variable.

Here we can see some p-hat values which show the probability of taking up to the offer for different sample age groups.

As a matter of fact, we draw this line of probability = 0.5 (in most of the cases), and map all points below to this value to y^ = 0; and for all values above it to y^ = 1 i.e. they will take this offer.

Logistic Regression



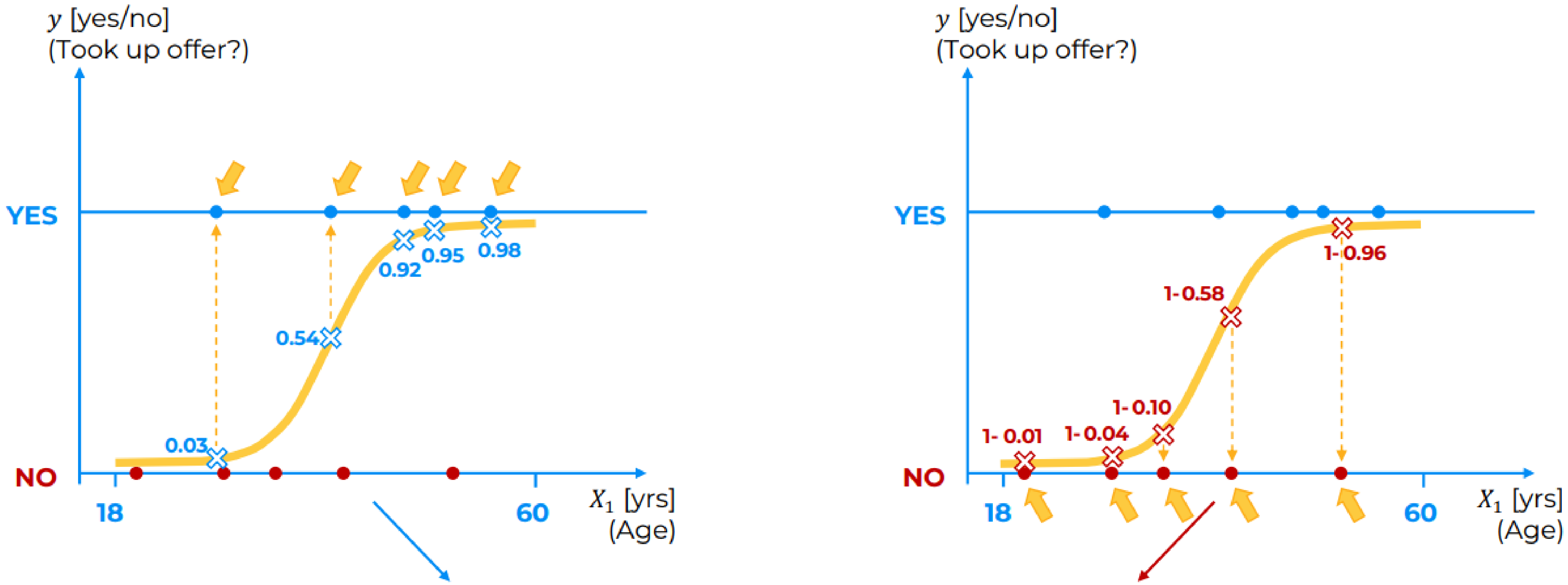
Scientific Explanation for Logistic Regression

It is pertinent to understand that it works like linear regression, i.e. we agree on a line and try to fit a best line for our input data, and try to draw inferences from this line.

We can calculate the probabilities for different events. We can also get the predicted value for the dependent variable based on where we select this arbitrary line i.e. 0.5 in this case.

We can place this line at different positions depending upon the nature of the problem and our domain knowledge to get the best predictions.

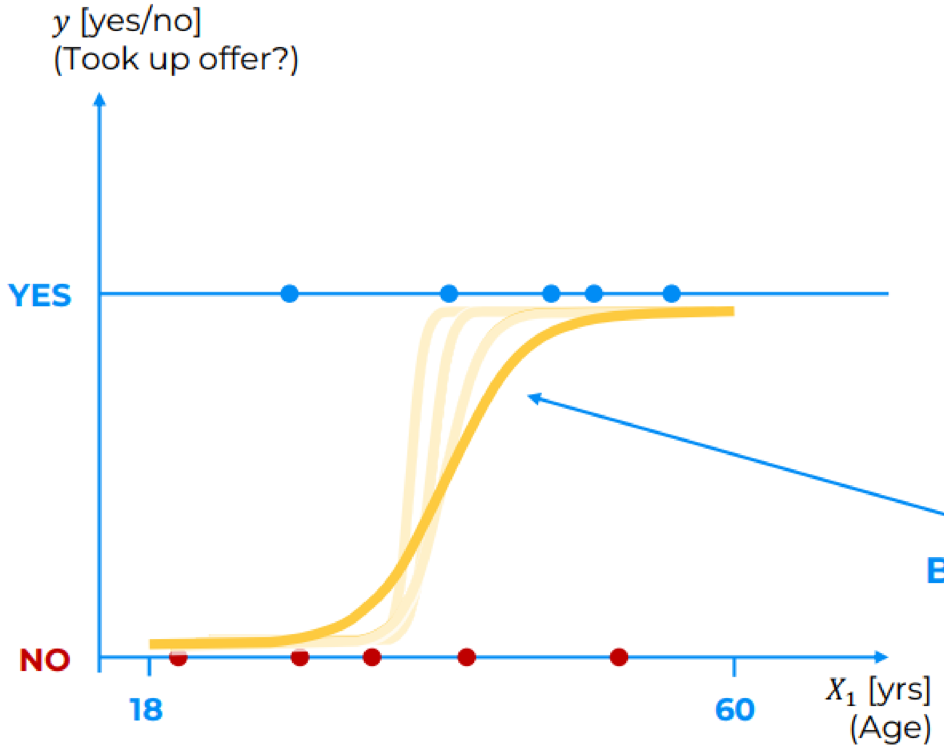
Maximum Likelihood



Likelihood = 0.03 x 0.54 x 0.92 x 0.95 x 0.98 x (1 – 0.01) x (1 – 0.04) x (1 – 0.10) x (1 – 0.58) x (1 – 0.96)

Likelihood = **0.00019939**

Maximum Likelihood



Likelihood = 0.00007418

Likelihood = 0.00012845

Likelihood = 0.00016553

$\boxed{\text{Likelihood = 0.00019939}}$

Best Curve <= Maximum Likelihood

Sample Example and Implementation

- Purchase SUV or Not
- Features
  - Age
  - Estimated Salary
- Disregard feature
  - User id
  - Gender

	A	B	C	D	E	F
1	User ID	Gender	Age	Estimated	Purchased	
2	15624510	Male	19	19000	0	
3	15810944	Male	35	20000	0	
4	15668575	Female	26	43000	0	
5	15603246	Female	27	57000	0	
6	15804002	Male	19	76000	0	
7	15728773	Male	27	58000	0	
8	15598044	Female	27	84000	0	
9	15694829	Female	32	150000	1	
10	15600575	Male	25	33000	0	
11	15727311	Female	35	65000	0	
12	15570769	Female	26	80000	0	
13	15606274	Female	26	52000	0	
14	15746139	Male	20	86000	0	
15	15704987	Male	32	18000	0	
16	15628972	Male	18	82000	0	
17	15697686	Male	29	80000	0	
18	15733883	Male	47	25000	1	
19	15617482	Male	45	26000	1	
20	15704583	Male	46	28000	1	

```python
# Logistic Regression

# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('Social_Network_Ads.csv')
#we intend to make classification decision based on age and salary parameters only
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, -1].values

# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

# Feature Scaling
#in this example we need feature scaling because the value ranges differ significantly
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Training the Logistic Regression model on the Training set
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)

```

```python
# Logistic Regression

# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('Social_Network_Ads.csv')
#we intend to make classification decision based on age and salary parameters only
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, -1].values

# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

# Feature Scaling
#in this example we need feature scaling because the value ranges differ significantly
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Training the Logistic Regression model on the Training set
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

# Making the Confusion Matrix
#confusion matrix presents correctly classified data and incorrectly classified data for different classes
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

Name	Type
X	int64
X_test	float64
X_train	float64
cm	int64
dataset	DataFrame
y	int64
y_pred	int64
y_test	int64

Variable explorer | File

IPython console

Console 1/A

```
verbose=0,

In [6]: y_pred = cl

In [7]: from sklear
   ...: cm = confus
   ...: print(cm)
[[65  3]
 [ 8 24]]

In [8]:
```

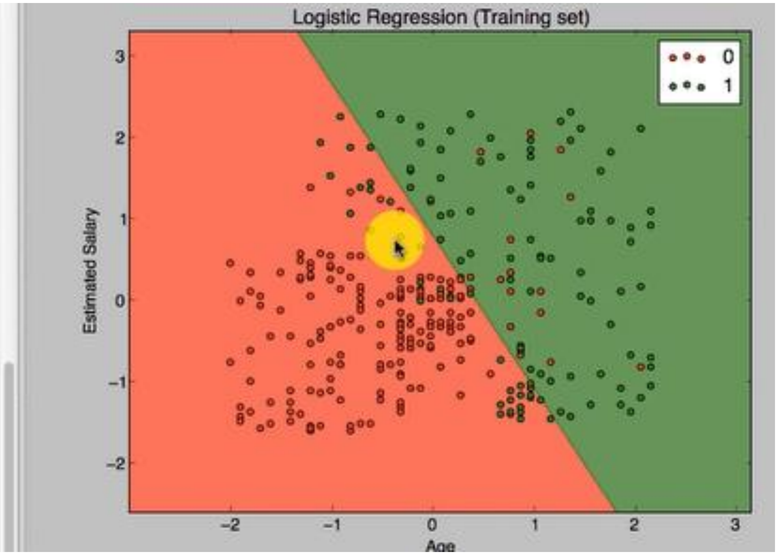
```python
# Predicting the Test set results
y_pred = classifier.predict(X_test)

# Making the Confusion Matrix
#confusion matrix presents correctly classified data and incorrectly classified data for different classes
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)

# Visualising the Training set results
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Logistic Regression (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

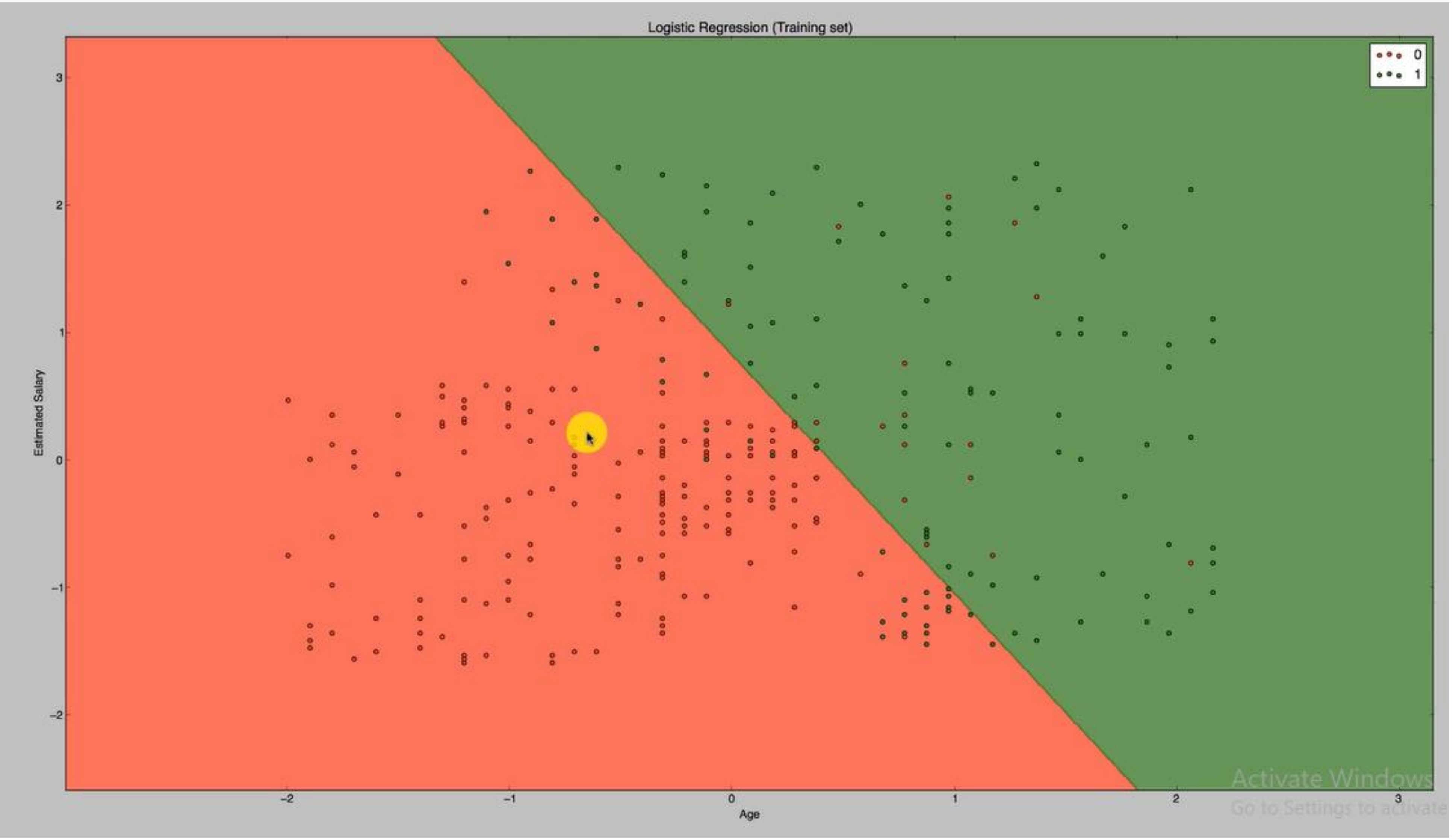
```
   ...: print(cm)
[[65  3]
 [ 8 24]]
```

```
34
35 # Visualising the Training set results
36 from matplotlib.colors import ListedColormap
37 X_set, y_set = X_train, y_train
38 X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
39                      np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
40 plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
41              alpha = 0.75, cmap = ListedColormap(('red', 'green')))
42 plt.xlim(X1.min(), X1.max())
43 plt.ylim(X2.min(), X2.max())
44 for i, j in enumerate(np.unique(y_set)):
45     plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
46                 c = ListedColormap(('red', 'green'))(i), label = j)
47 plt.title('Logistic Regression (Training set)')
48 plt.xlabel('Age')
49 plt.ylabel('Estimated Salary')
50 plt.legend()
51 plt.show()
```



Line 37: X_set, y_set are two local variables which help using the same code for plotting the test set data with line X_set, y_set = X_test, y_test.
Lines 38-39: Plots pixel points with resolution 0.01 (step variable). Actually, with this code it is using the classifier to plot all the pixel points with different values of Age and Estimated Salary (the two variables) and draws the red and green regions for all the hypothetical values with 0.01 pixel density. Thus the grid is prepared.
Min -1 and Max +1 are used for both variables Age and Estimated Salary so that the plotted points should have some distance from the grid boundary.
Lines 40-41: Contour function is used to draw this contour line that is marking the boundary between the two regions.
Lines 42-43: Plots the limits of the x and y axis. (X -> age, y -> estimated Salary

Lines 44-46: The loop plots all the data points in the form of a scatter plot.
Rest of the lines show labels on x and y axes, legend (red and green for 0 and 1 class), and display the plot.



<u>TRAINING SET PLOT</u>

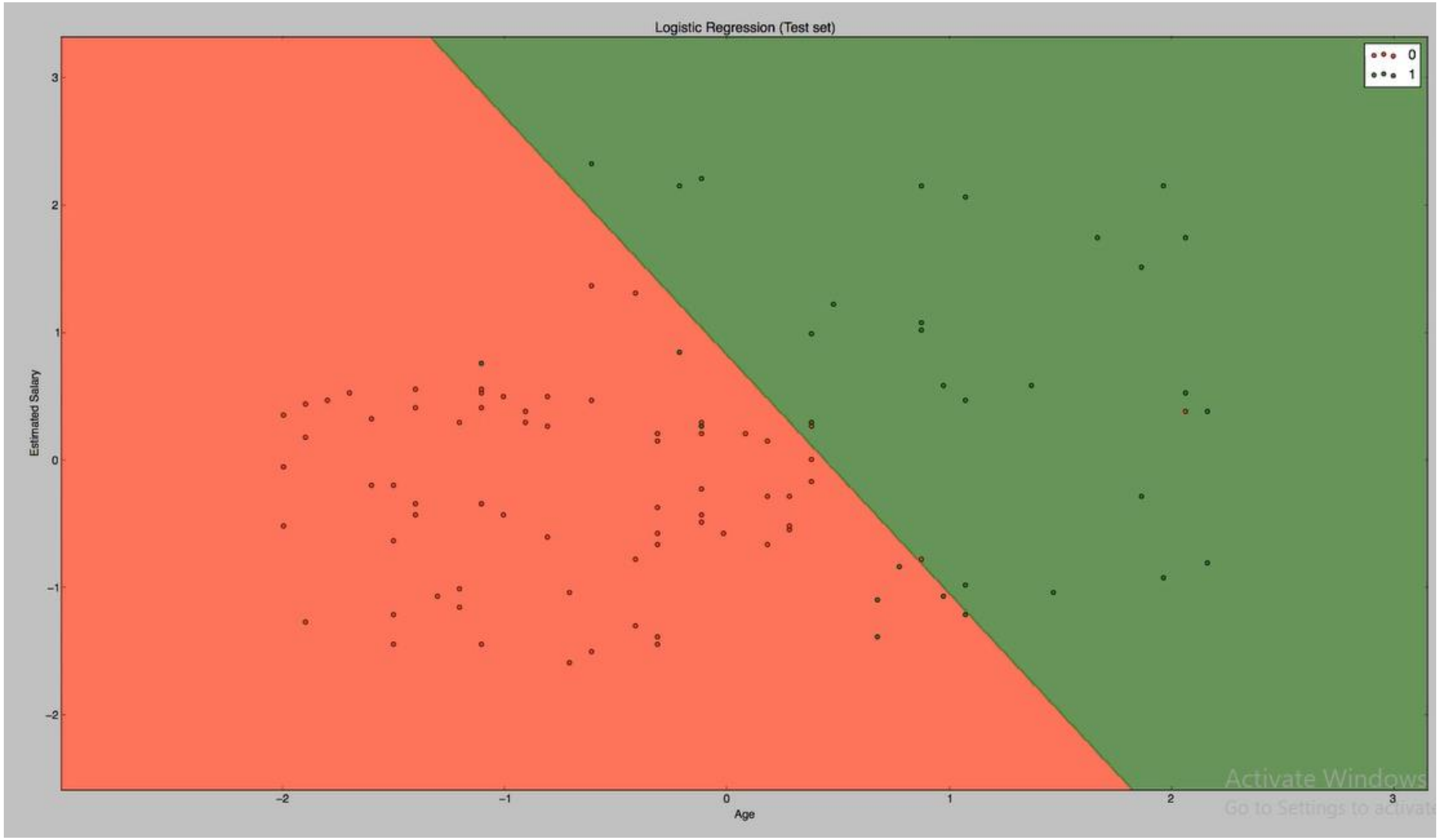
Boundary is a straight line, as logistic regression is a linear classifier. In higher dimensions it will be plane or hyperplane.

The regions are well fitted according to the training data set, though we have some incorrect plots.

Thus, fulfilling the goad of plotting right users in right categories.

```python
# Visualising the Test set results
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Logistic Regression (Test set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

X_set, y_set are two local variables which help using the same code for plotting the test set data with line X_set, y_set = X_test, y_test



TEST SET PLOT
Test set is reflecting the result of the confusion matrix.