upGrad

#LifeKoKaroLift

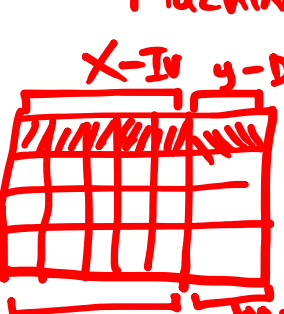
Logistic Regression

Module : Logistic Regression

Instructor: Dr. Darshan Ingle

upGrad

Machine Learning (designing a Predictive Model)



Supervised

X & y → Given

100 × 6

X_train, y_train

X_test, y_test

import LR

lr = LR()

model = lr.fit(X_train, y_train)

y_pred = model.predict(X_test)

mse(y_pred, y_test)

Classification
(predict → Categorical value)

Regression (y → continuous)
(predict → Continuous value)

Unsupervised

X → Given

Role: Decide/Define/Find/Label the appropriate T.U.

eg: KMeans, Apriori, etc.

Reinforcement Learning

[60–80 hrs]

- His learning from own observation
- Reward & Penalty
- SCORE = 1000

Tesla Auton. Car (USA) → No speed breaker

25/50 is 5/10

Score = 1000 + 1 + 1 − 10 − 10 − 10 + 1 + 1 + 1 + 1 ...

Binary Classification

3 yrs old → Shapes

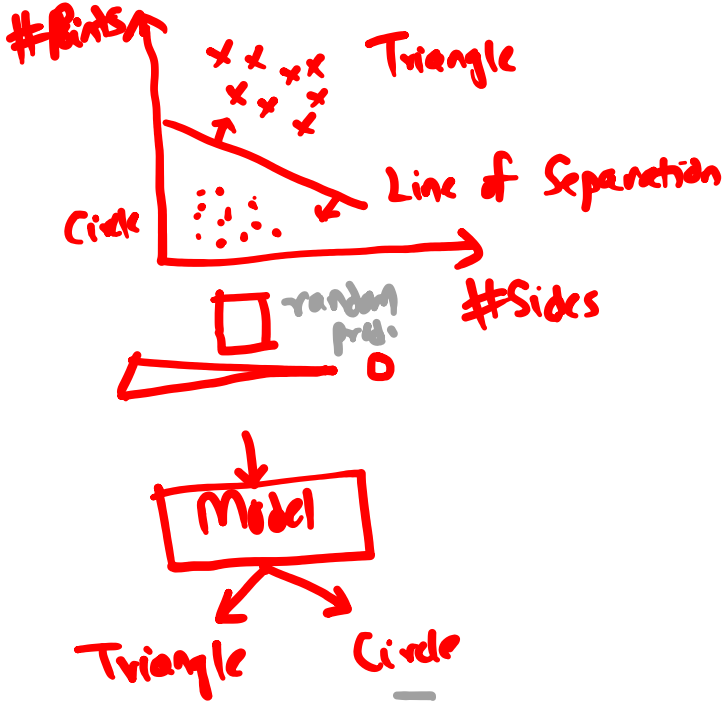
QB AB

X	y
△	Triangle
○	Circle
◁	Triangle
○	Circle
⋮	⋮

100 x 2

(50 - △, 50 - ○)

	f1 Sides	f2 Points
△	3	3
○	0	0



Note: All ML problems work on principle "GIGO".

Classification Types:

(a) Linear Classifiers:
1. Logistic Regression
2. Naive Bayes Classification

(b) Support Vector Machines:
1. Support Vector Classifier

(c) Decision Trees
1. DT Classifier (Overfitting)

(d) Random Forest
1. RF Classifier

(e) Boosted Trees
1. Gradient Boosting Classifier
2. AdaBoost —1—
3. XGBoost —4—

(f) ~~Naive~~ Nearest Neighbor
1. KNN Classifier

(g) Neural Network

[Parametric ML]

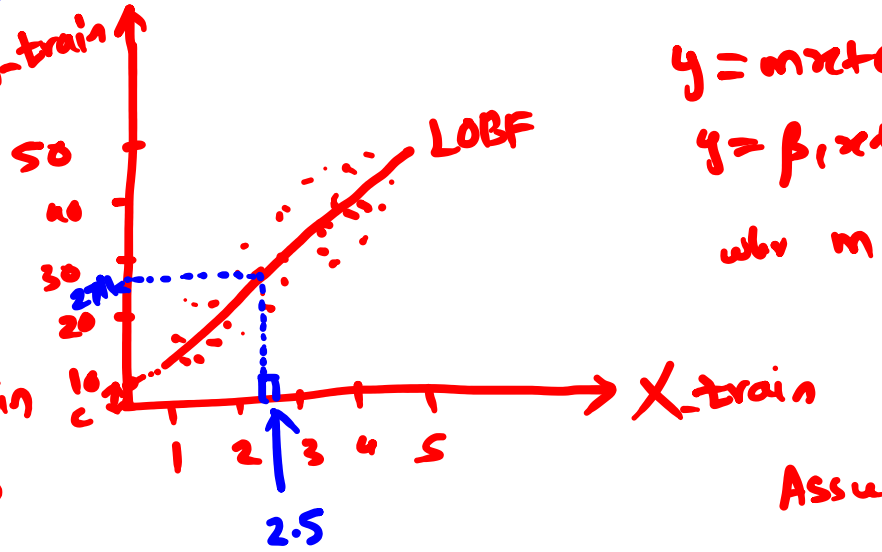
Linear Regression:

AB(X)	AB(y)
Rooms	Price
1	10
2	20
3	30
4	40
5	50
4	42
4	37
1	12
⋮	⋮

X_train — Train — y_train

X_test — Test — y_test

100 x 2



$y = mx + c$ or

$y = \beta_1 x + \beta_0$

where $m = \text{Slope} = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1}$

c = y-intercept

Assume, $m = 0.9$

$c = 2$

$\therefore y = 0.9x + 2$

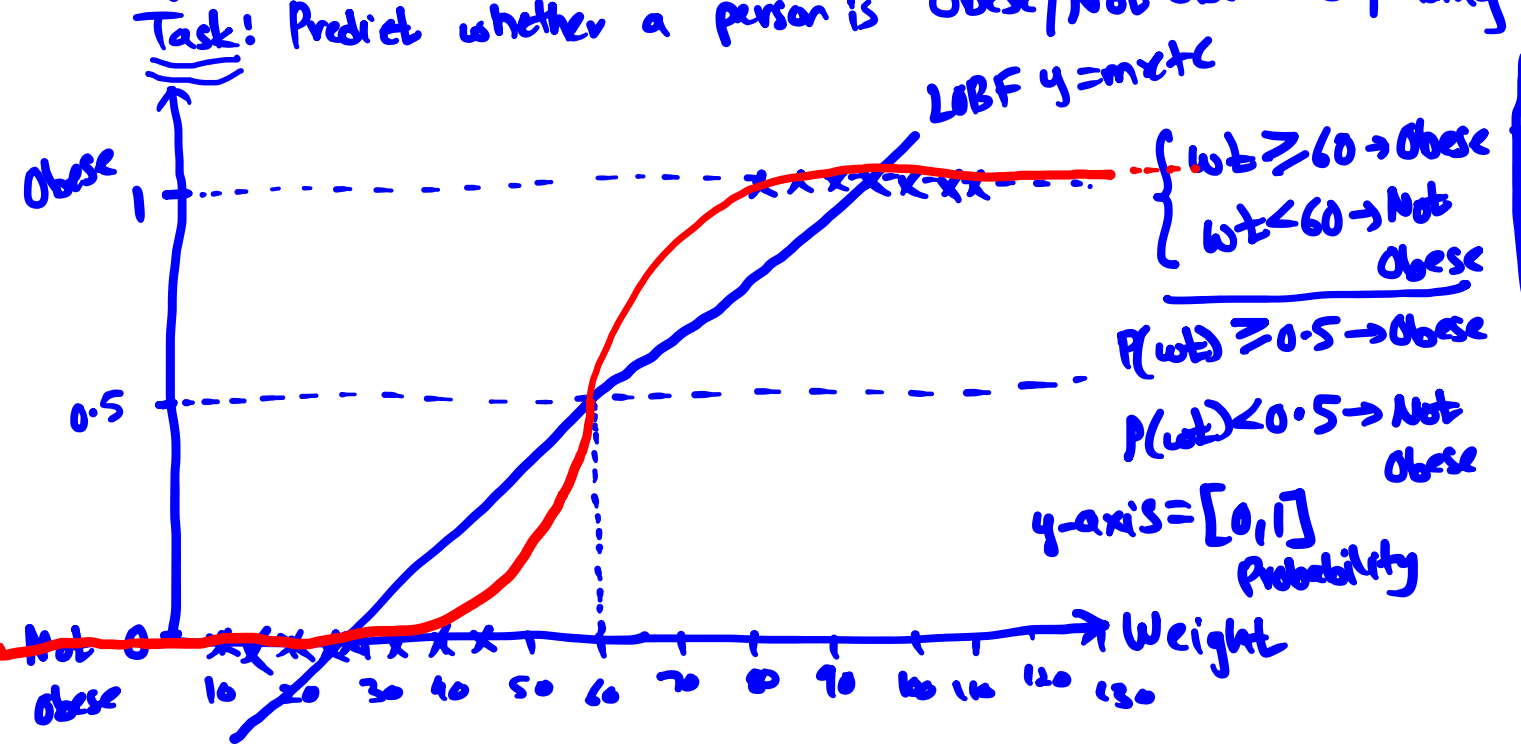
$y = 0.9(2.5) + 2$

$y = 27$ lakh

Logistic Regression: Used for ① Binary Classification (Industry) ② Multi-class Classification

Binary Classification

Task: Predict whether a person is Obese/Not Obese depending on weight.



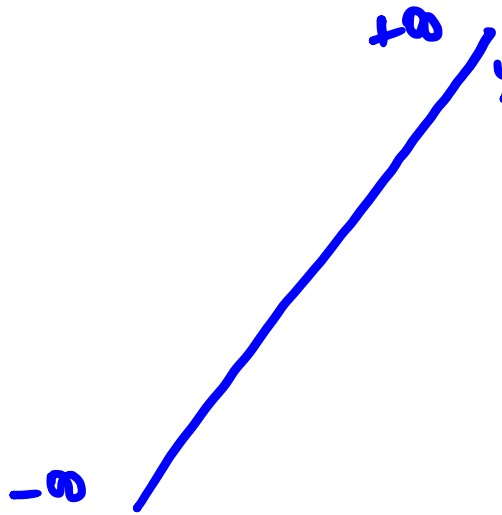
X: wt	y: Class
10	Not Obese
20	Not Obese
80	Obese
35	N. Obese
78	Obese
⋮	⋮

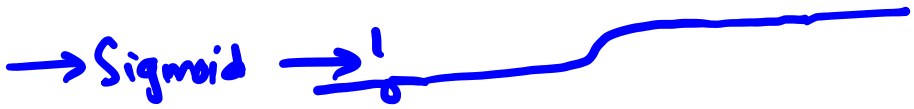
Sigmoid: $\sigma = \frac{1}{1+e^{-y}}$, where $y = mx+c$ or $y = \beta_1 x + \beta_0$

$\therefore \sigma = \frac{1}{1+e^{-(mx+c)}}$

$\sigma \rightarrow$ Squashes it $[0,1]$

$[-\infty, +\infty] \Rightarrow [0,1]$





Logit Function:

P = Prob. of Success

$$\log\left(\frac{P}{1-P}\right) = mx + c$$

$$\therefore \boxed{\frac{P}{1-P} = e^{mx+c}}$$

Logit or Log-Odd Function.

$$\log\left(\frac{P}{1-P}\right)$$

$\Downarrow$

log - logarithm (ln)

$\left(\frac{P}{1-P}\right)$ = Odds

What are Odds?

Odds → Signifies the ratio of Prob(Success) to Prob(Failure)

$$\text{Odds} = \frac{P(\text{Success})}{P(\text{Failure})} = P\left(\frac{\text{Success}}{\text{Failure}}\right) = P\left(\frac{0.7}{1-0.7}\right)$$

$$= \text{Prob}\left(\frac{0.7}{0.3}\right) = 2.3333$$

These are odds in favor.

Pros:

1. Simple
2. Efficient
3. Low Variance
4. Provides Probability Score for every observation

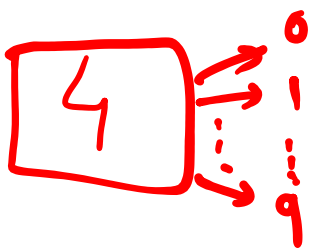
   → predict_proba()

Cons:

1. Doesn't handle large no. of Categorical features.
2. Requires Transformation of Non-linear features.

Applications!

1. Image Segmentation.
2. Categorization
3. Object Detection
4. Geographical Image Processing
5. Handwritten Character Recognition



Multi-class Classification: One vs Rest Approach

3 Classifier

0.20 1. Dot vs Rest — Classf#1

0.21 2. Cross vs Rest — Cl.2

→ 0.72 3. Triangle vs Rest → Cl.3

△ — Triangle

upGrad

#LifeKoKaroLift

Thank You!

Connect me: https://www.linkedin.com/in/dr-darshan-ingle-corporate-trainer/