Logistic Regression

Machine Learning (CS 306)

Instructor: Dr. Moumita Roy

Teaching Assistants: Indrajit Kalita, Veronica Naosekpam

Email-ids:

moumita@iiitg.ac.in

veronica.naosekpam@iiitg.ac.in

indrajit.kalita@iiitg.ac.in

Mobile No: +91-8420489325 (only for emergency quires)

Why use logistic regression?

- There are many important research topics for which the dependent variable is "limited."
- For example: voting, spam or not spam, and participation data is not continuous.
- Binary logistic regression is a type of regression analysis where the dependent variable is a dummy variable: coded as 0 (did not vote) or 1(did vote)

Classification

- Email: Spam/not spam
- Land-cover: Water-cover/not water-cover
- Customer Behavior Prediction: Sad/Happy
- Tumor: Malignant/not Malignant

$Y \in \{0,1\}$

- coded as binary dependent variable (two-class problem)

  0→Negative class

  1 →Positive class

Linear Regression (Revisit)

$$x_i = [x_{i,1} \quad x_{i,2} \quad x_{i,3} \quad x_{i,4} \quad \cdots \quad x_{i,n}]$$

$$h(x_i) = \sum_{j=0}^{n} w_j x_{i,j} \qquad x_{i,0} = 1$$

$$w_j = [w_0 \quad w_1 \quad w_2 \quad \cdots \quad w_n]$$

$$h(x_i) = w^T x_i$$

$$J(w) = \frac{1}{2m} \sum_{i=1}^{m} (h(x_i) - y_i)^2$$

↓ cost function

$$w_j := w_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h(x_i) - y_i)\, x_{i,j}$$

↓ parameter updation using GD

Important Points

- Can we use solution of linear regression problem directly for classification?

- It may be formulated like this (threshold the output):

$h(X_i) >= 0.5$, predict Y_i as 1

$h(X_i) < 0.5$, predict Y_i as 0

- Problem in the present approach:

I. The output $h(X_i)$ is unbounded

$h(X_i) > 1$ or < 0

II. In this case, fixing threshold at 0.5, is reasonable or not?

Solution: search a function which can bound the output between 0 to 1

Logistic Regression (Classification)

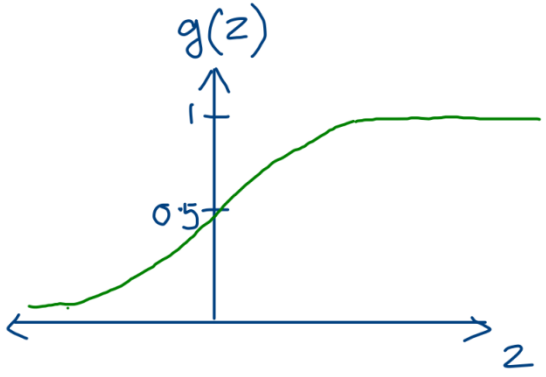
- Aim is to range the output: $0 <= h(X_i) <= 1$

$$h(x_i) = g(w^T x_i)$$

- $g(...)$ as sigmoid function/logistic function

$$g(z) = \frac{1}{1+e^{-z}}$$

$$h(x_i) = \frac{1}{1+e^{-w^T x_i}}$$



$g(z) \to 0$ if $z \to -\infty$

$g(z) \to 1$ if $z \to \infty$

Interpretation of Hypothesis output

- $h(X_i)$=estimated probability that $Y_i=1$ for a given X_i
- Suppose, $h(X_i)=0.7$

  interpret as, there is a 70% chance that X_i belongs to $Y_i=1$; alternatively there is a 30% chance that X_i belongs to $Y_i=0$ (if estimate in the scale of 100)

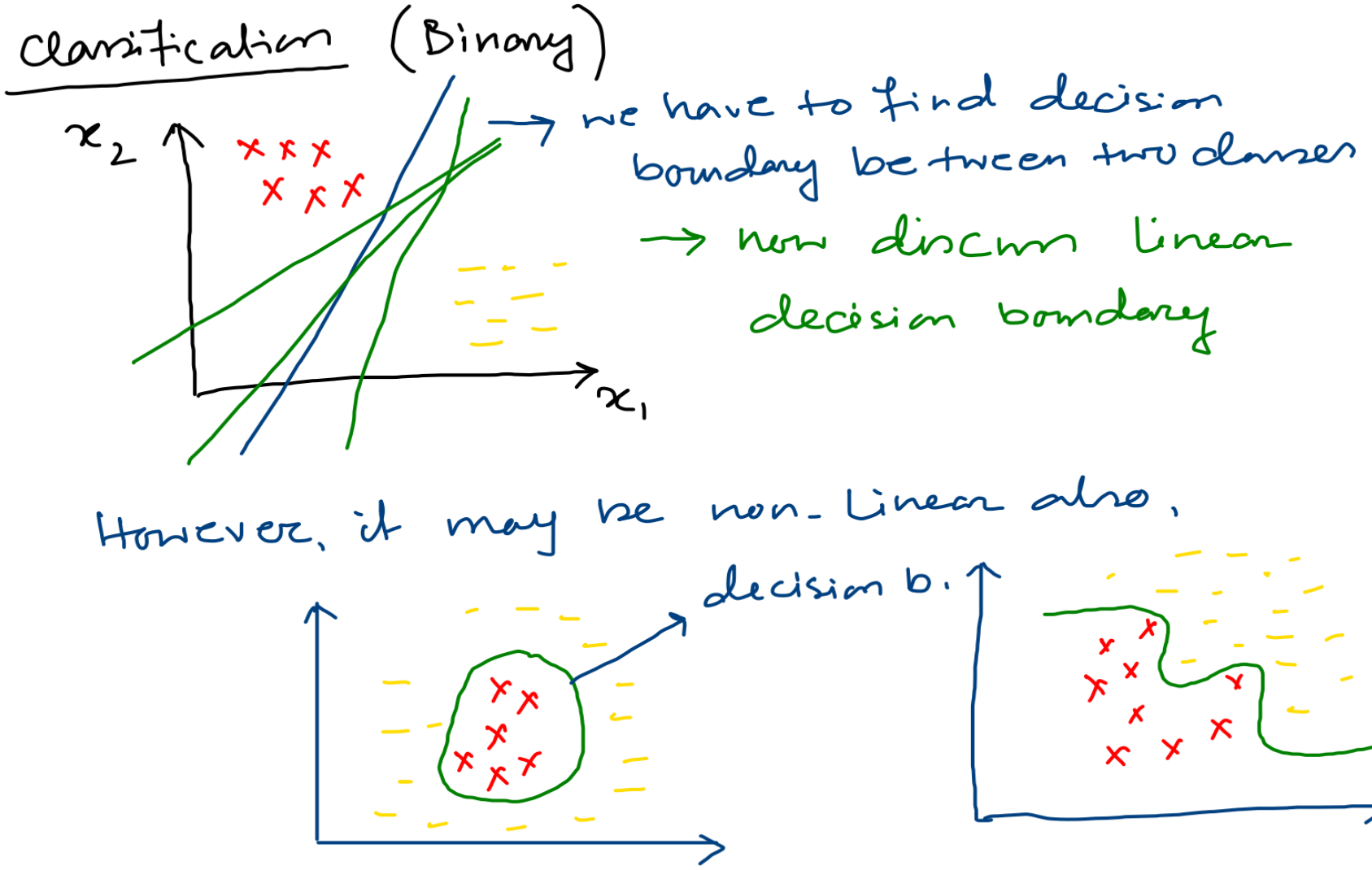
- $h_w(X)=P(Y=1|X;W)$ [probability that $Y=1$ for given X; parameterized by W]

- Since, Y is either 0 or 1

$P(Y=1|X;W)+ P(Y=0|X;W)=1$

$P(Y=0|X;W)=1-P(Y=1|X;W)$

Classification (Decision Boundary)



Simple case (two features)

$ax + by = c$

DB : Line

$w_1 x_1 + w_2 x_2 = w_0$

Aim: Find the values of parameters

$w = [w_0\ w_1\ w_2]$ using training

Samples $\langle x_i, y_i \rangle$

$w_1 x_1 + w_2 x_2 - w_0 \geq 0$ (One side of line)

↳ for such samples $Y = 1$ (positive class)

$w_1 x_1 + w_2 x_2 - w_0 < 0$ (Other side of line)

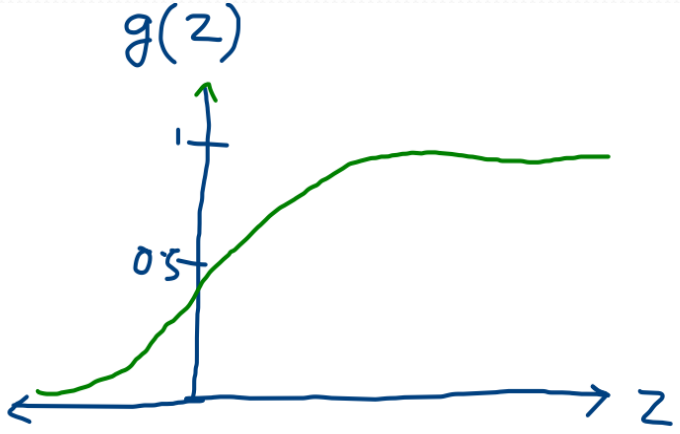
↳ for such samples $Y = 0$ (negative class)

Logistic Regression

$h(x_i) = g(w^T x_i)$

$h(x) = g(w^T x)$

$g(z) = \frac{1}{1+e^{-z}}$



→ $g(z) \geq 0.5$ when $z \geq 0$

Suppose, predict $y=1$; $h(x) \geq 0.5$

$w^T x \geq 0$ [Same as the previous example for two feature case]

predict $y=0$; $h(x) < 0.5$

$w^T x < 0$

Data set for classification

x_1	x_2	y
2	0.6	spam
1	5	not spam
6	2.8	not spam
3	1.4	spam

Format 1

class label
1
2
2
1

Format 2

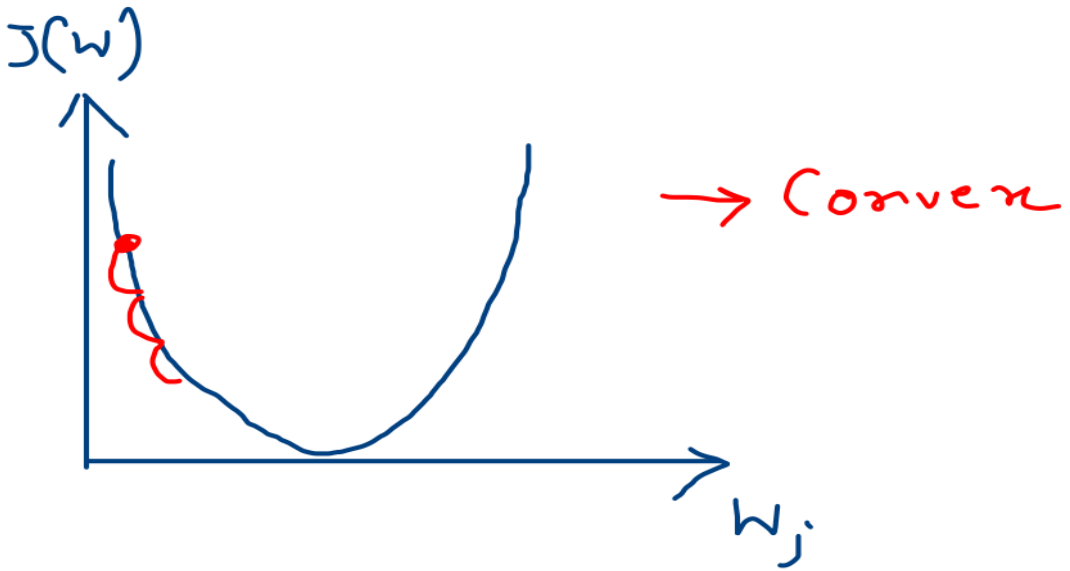
y
0
1
1
0

After re-write the class label

Logistic Regression (Cost Function)

Linear regression

$$J(w) = \frac{1}{2m} \sum_{i=1}^{m} \left(h(x_i) - y_i \right)^2$$



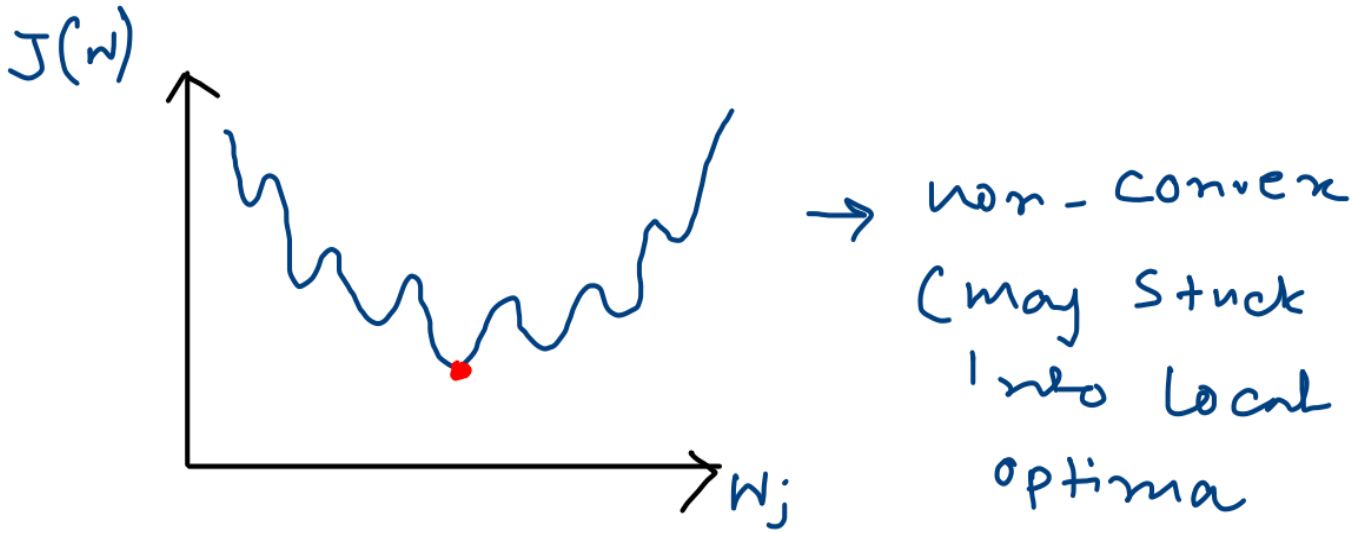
Logistic Regression

$$J(w) = \frac{1}{2m} \sum_{i=1}^{m} \left(h(x_i) - y_i\right)^2$$ → Is it convex included here

$$h(x_i) = g(w^T x_i)$$

$$= \frac{1}{1 + e^{-w^T x_i}}$$

} non-linearity involves



→ non-convex (may stuck into local optima

Logistic regression with MSE

$$J(w) = \frac{1}{2m} \sum_{i=1}^{m} \left(h(x_i) - y_i \right)^2$$

$$h(x_i) = g(w^T x_i)$$

$$g(z) = \frac{1}{1+e^{-z}}$$

$$g'(z) = g(z)\left(1 - g(z)\right)$$

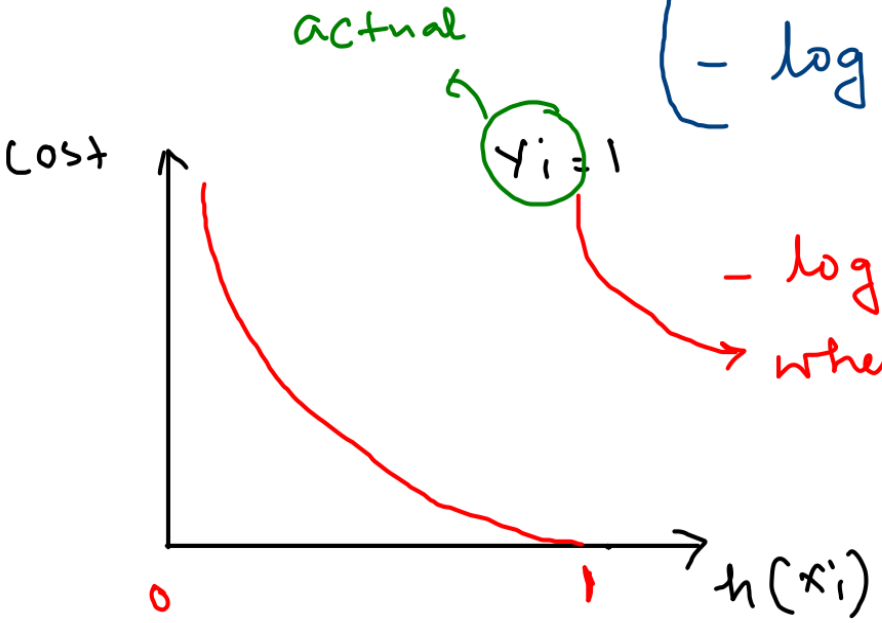
SGD: $w_j := w_j - \frac{\partial}{\partial w_j} J(w)$

$$w_j := w_j + \alpha \left(y_i - h(x_i) \right) x_{i,j} \underbrace{h(x_i) \cdot \left(1 - h(x_i)\right)}$$

Cost function for logistic regression

$$J(w) = \frac{1}{m} \sum_{i=1}^{m} \text{cost}\left(h(x_i), y_i\right)$$

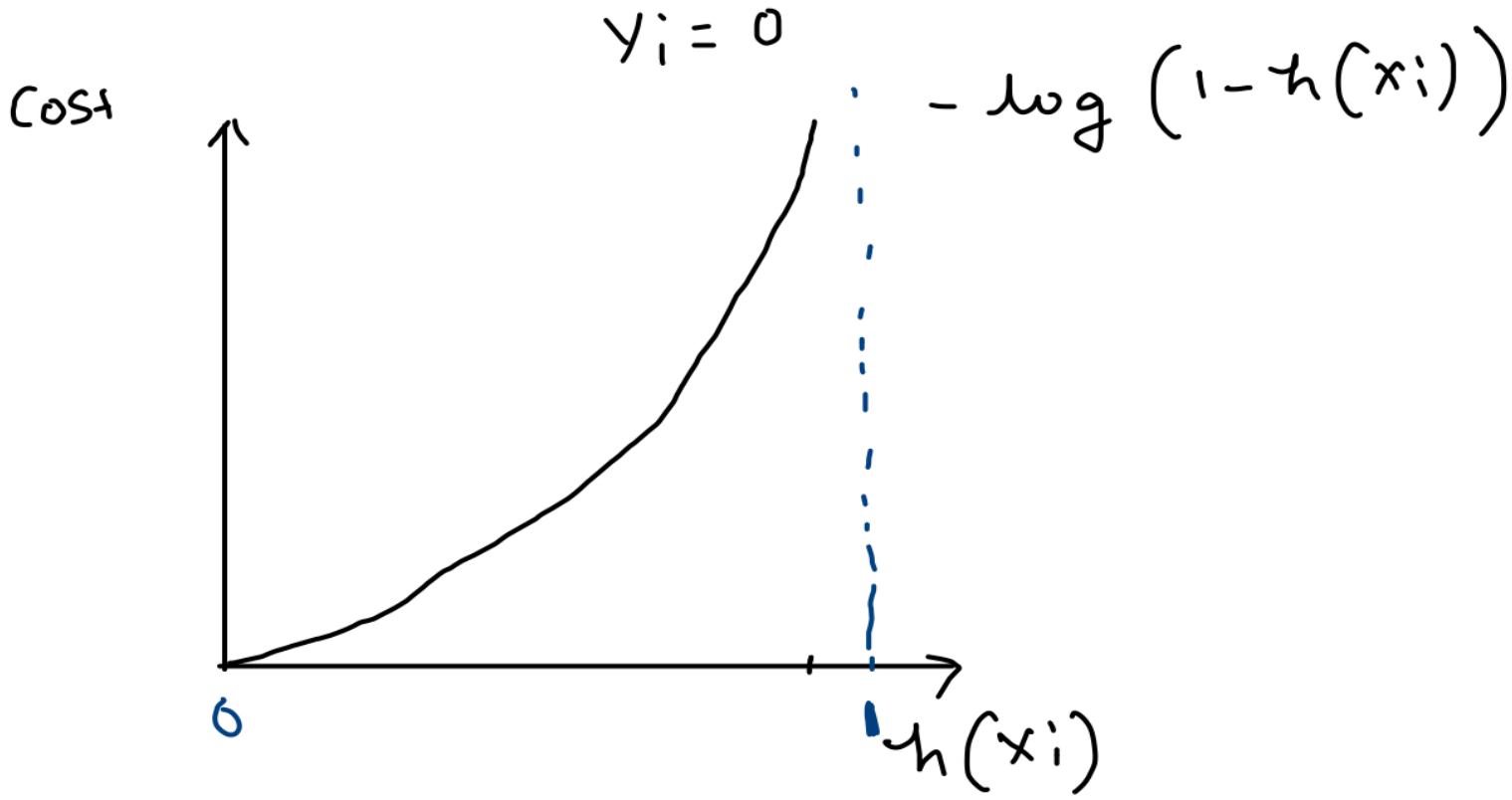
$$\text{Cost}\left(h(x_i), y_i\right) = \begin{cases} -\log\left(h(x_i)\right) & \text{if } y_i = 1 \\ -\log\left(1 - h(x_i)\right) & \text{if } y_i = 0 \end{cases}$$



$-\log(h(x_i))$ when, $h(x_i) = 1$ (predicted)

$\text{cost}(h(x_i), y_i) = 0$ if $h(x_i) \to 0$

cost $\to \infty$



if $y_i = 0$, $h(x_i) = 0$, Cost = 0

$h(x_i) \to 1$

Cost $\to \infty$

Simplified Cost function

$$\text{Cost}\left(h(x_i), y_i\right)$$

$$= -y_i \log\left(h(x_i)\right) - (1-y_i) \log\left(1 - h(x_i)\right)$$

$$y_i = 0 \text{ or } 1$$

when $y_i = 0$

$$\text{Cost}\left(h(x_i), y_i\right) = -\log\left(1 - h(x_i)\right)$$

when $y_i = 1$

$$\text{Cost}\left(h(x_i), y_i\right) = -\log\left(h(x_i)\right)$$

Cost function for logistic regression

$$J(w) = \frac{1}{m}\sum_{i=1}^{m} \text{Cost}\left(h(x_i), y_i\right)$$

$$= \frac{1}{m}\sum_{i=1}^{m}\left[-y_i \log\left(h(x_i)\right) - (1-y_i)\log\left(1-h(x_i)\right)\right]$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left[y_i \log\left(h(x_i)\right) + (1-y_i)\log\left(1-h(x_i)\right)\right]$$

→ minimize $J(w)$ to obtain

$w = [w_0\ w_1\ \ldots\ w_n]$

$h(x_i) = \frac{1}{1+e^{-w^T x_i}}$

GD: $w_j := w_j - \alpha\left(h(x_i) - y_i\right) x_{i,j}$

→ Same updation rule as linear regression

Algorithmic representation for gradient descent

Step 1: Randomly initialize $w_0, w_1, ..., w_n$ and learning rate

Step 2: Calculate $h(X_i)$ for all training samples

Step 3: Calculate $J_{itr}(W)$ for itr^{th} epoch

Step 4: Update all parameters $w_0, w_1, ..., w_n$

Step 5: Repeat steps 2-5 until

$$| J_{itr}(W) - J_{itr-1}(W)| > \rho \text{ or number of epochs} > no$$

Step 6: Apply threshold to take final decision for binary classification

Step 6: Report the value of $J_{itr}(W)$ and classification accuracy

ρ: small value (may be *0.0001*)

***no*: sufficiently large so that model can learn**

Can we stop using validation set?

Epoch/training step: one time updating considering all the training samples

GD for Logistic regression

Training samples

x_1	x_2	Class Label
1.2	3	1
3	4	2
6	7	1
9	10	1

$W = [1 \quad 1 \quad 1]$

$\alpha = .1$

$$g(z) = \frac{1}{1+e^{-z}}$$

1. Normalized the dataset
2. Split train set / Test set / VS

TS

x_0	x_1	x_2	y
1	1.2	3	1
1	3	4	0
1	6	7	1
1	9	10	1

Epoch 1 $h(x_i) = g(w^T x_i) = g(w_0 x_0 + w_1 x_1 + w_2 x_2)$

TS1 : $h(x_1) = g(w_0 \times 1 + w_1 \times 1 \cdot 2 + w_2 \times 3)$

$= \cdot 6$ (not exact here / you need to calculate the exact one)

TS2 : $h(x_2) = \cdot 3$

TS3 : $h(x_3) = \cdot 5$

TS4 : $h(x_4) = \cdot 2$

Cost function → MSE (non-convex)

→ log loss (Convex)

MSE

$$J(w) = \frac{1}{2m} \sum (h(x_i) - y_i)^2$$

$$J(w) = \frac{1}{2m}\left[(\cdot 6 - 1)^2 + (\cdot 3 - 0)^2 + (\cdot 5 - 1)^2 + (\cdot 2 - 1)^2 \right]$$

$$w_j := w_j - \alpha \frac{\partial J(w)}{\partial w_j} \longrightarrow h(x_i) \text{ includes sigmoid function}$$

$$w_j := w_j + \alpha \frac{1}{m} \sum \left[(y_i - h(x_i)) x_{i,j} \right.$$

$$\left. h(x_i)(1 - h(x_i)) \right]$$

$$w_0 := w_1 + \alpha \frac{1}{m} \sum \left[\left((1 - \cdot 6) \times 1 \times \cdot 6 \times (1 - \cdot 6) \right) + \right.$$

$$w_1 := \quad (\quad w \quad) + (\quad t \quad)$$

$$w_2 := \quad + (\quad t \quad) \left. \right]$$

log loss

$$J(w) = -\frac{1}{m}\sum_{i=1}^{m}\left[y_i \log\left(h(x_i)\right) + (1-y_i)\log\left(1-h(x_i)\right)\right]$$

$$w_j := w_j - \alpha \frac{\partial J(w)}{w_j}$$

$$w_j := w_j + \alpha \frac{1}{m}\sum_{i=1}^{m}\left(y_i - h(x_i)\right) x_{i,j}$$

$w_0 = ?$

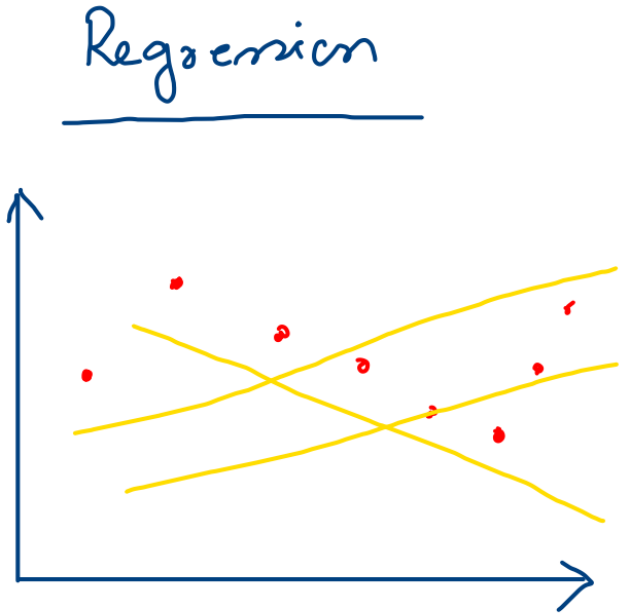
$w_1 = ?$

$w_2 = ?$

Repeat the training epoch by epoch until convergence (?)

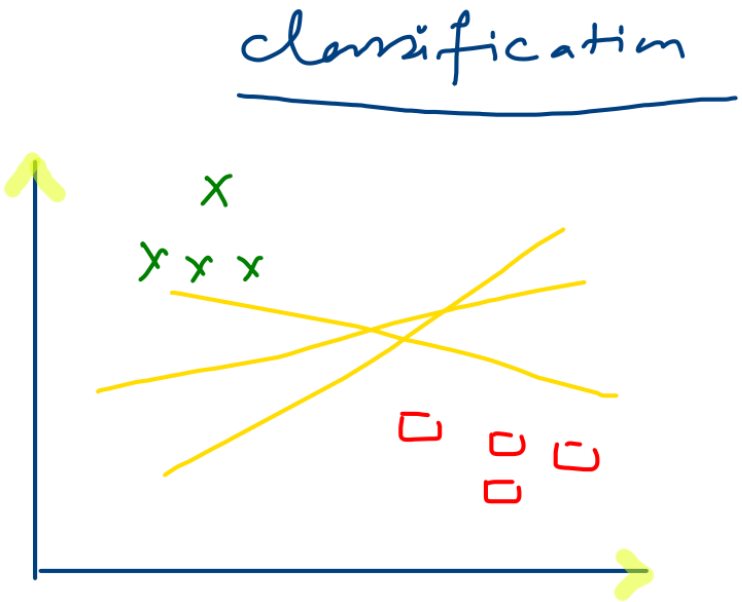
→ Finally report test accuracy after convergence of training

Regression vs. Classification



Linear Relation between independent and dependent variables

Try to find best fit one by minimizing cost function



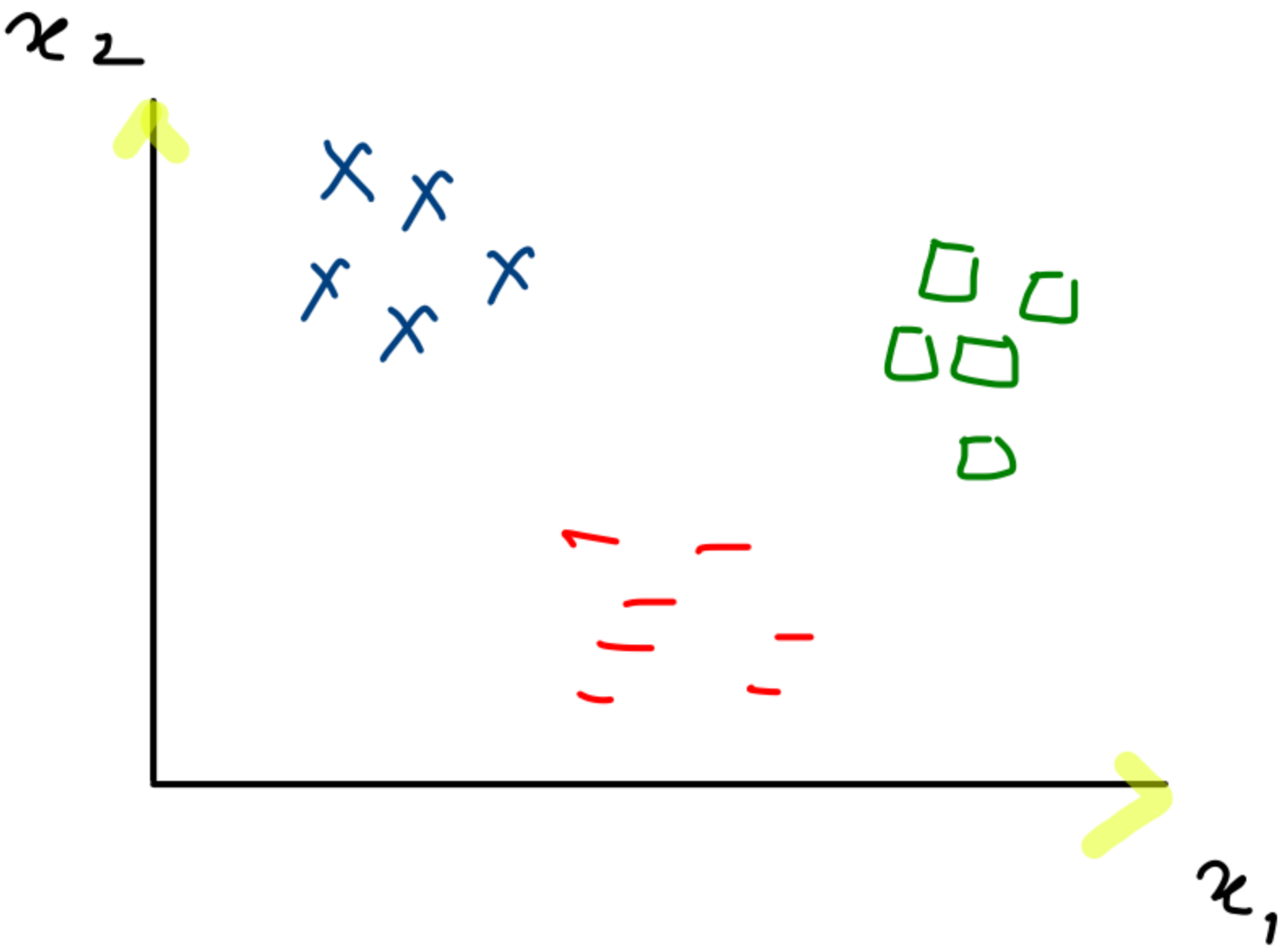
Data set is linearly separable

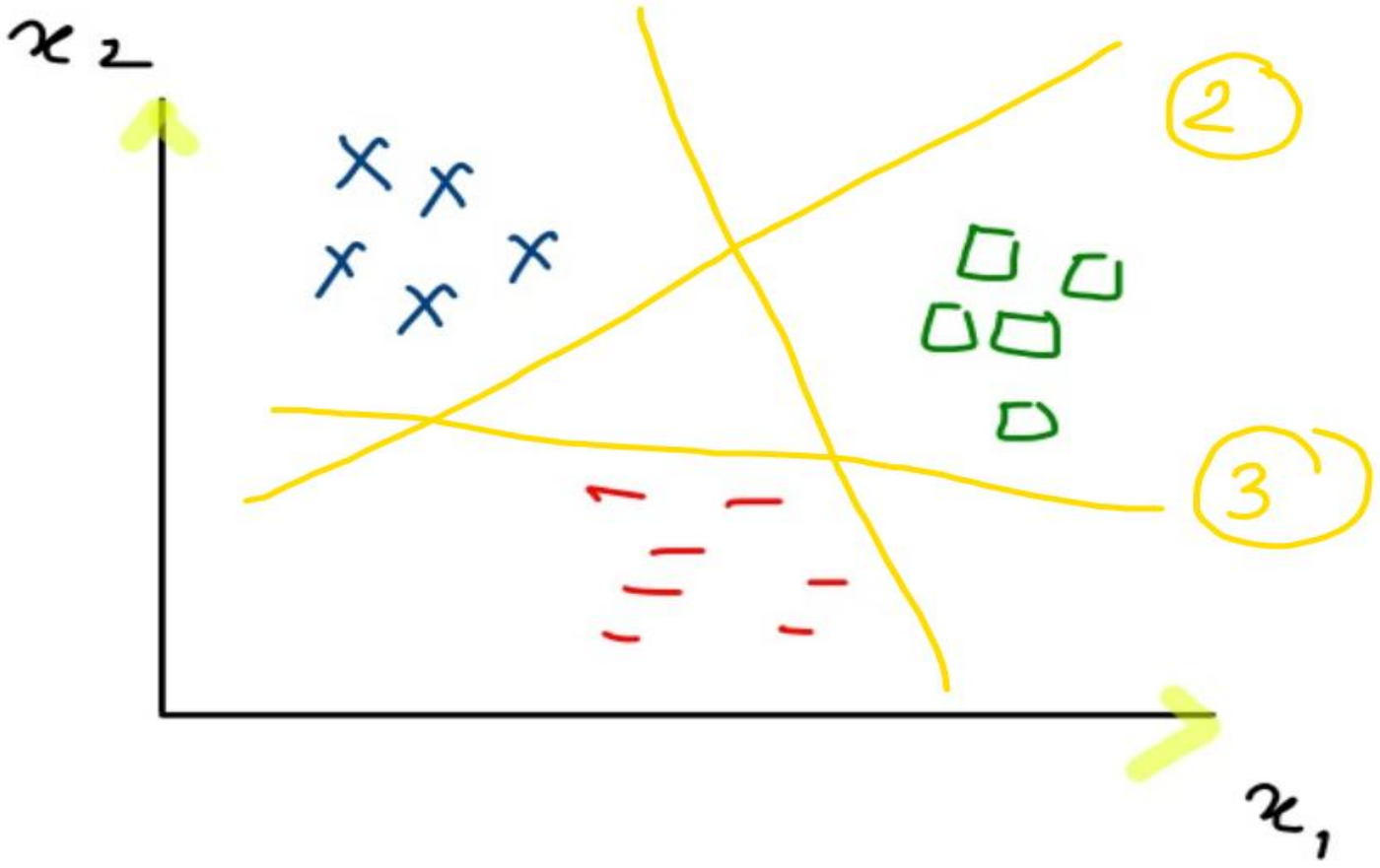
Try to find linear DB by minimizing cost function

Consider binary classification

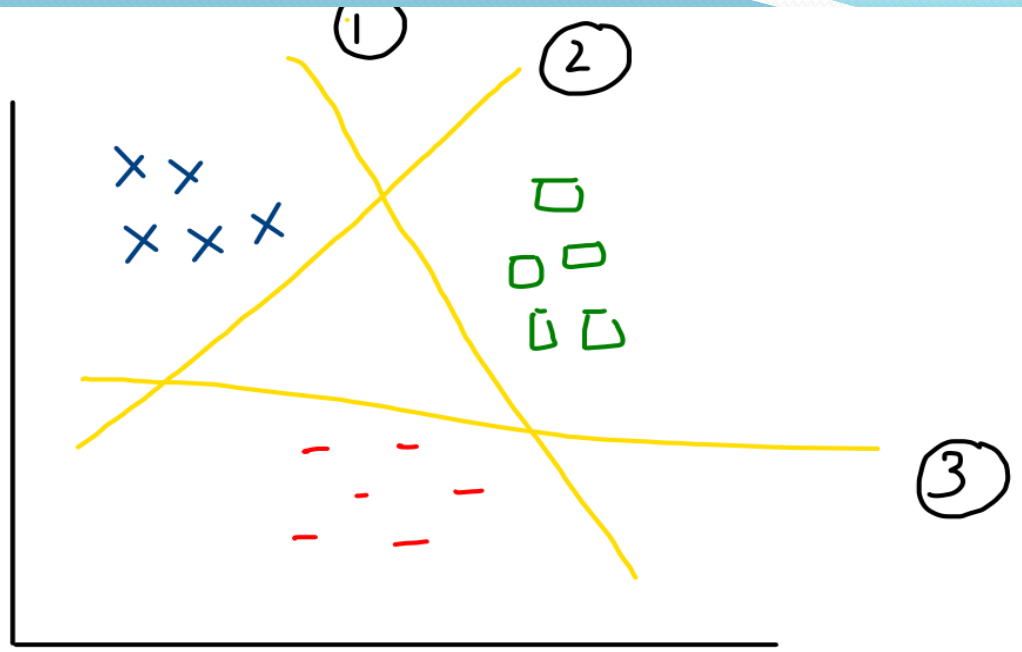
Multi-class Logistic Regression

x_1	x_2	class label	
2	3	1	
4	5	2	
6	7	3	→ Three classes
1·2	3	1	

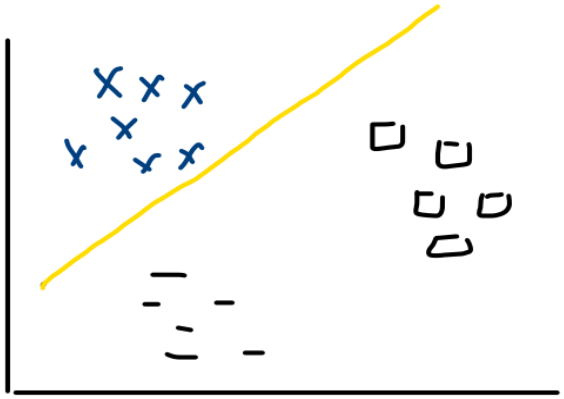

x_2
x_1



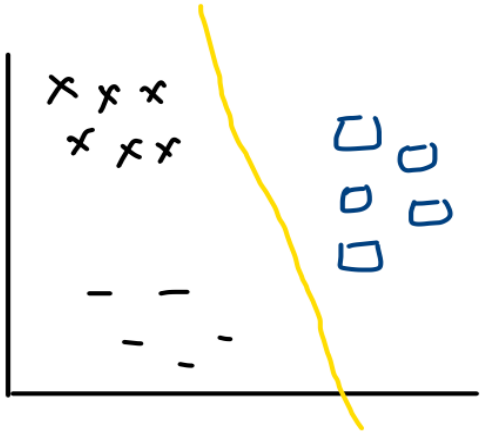
One vs. All



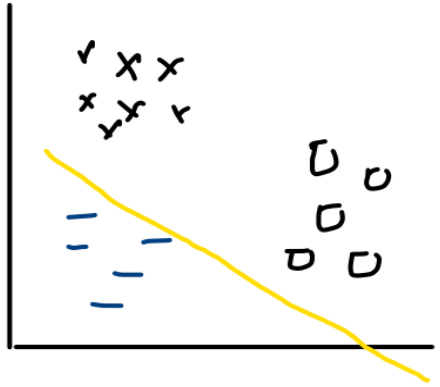
Multi-class, linearly separable



Classifier model 1



Classifier model 2



Classifier model 3

Dataset preparation??

Original Dataset (Training samples)

x_1	x_2	class label
2	3.4	1
6	1.2	2
3	4.5	2
6	1	3

Dataset to training classifier 1

x_1	x_2	Y
2	3.4	1
6	1.2	0
3	4.5	0
6	1	0

Same way prepare for classifier 2 and 3

m = number of pattern

$X_1 \quad X_2 \quad \cdots \quad X_i \quad \cdots \quad X_m$

n = number of features

$X_i = [x_{i,1} \quad x_{i,2} \quad \cdots \quad x_{i,j} \quad \cdots \quad x_{i,n}]$

c = number of classes

$k \in \{1, 2, \ldots, c\}$

One-vs-all

Train a logistic regression model $h^k(x_i)$ for each class k to predict probability that $y_i = k$ for a given x_i

→ Testing (Class label for unknown pattern)??

Testing phase

$x_i \rightarrow$ class label ?? $\rightarrow$ predicted class label in K

$h^1(x_i) \rightarrow$

$h^2(x_i) \rightarrow$

$h^3(x_i) \rightarrow$

$\vdots$

$h^K(x_i) \rightarrow$

$\vdots$

$h^C(x_i) \rightarrow$

$\Big\}$ – Choose K $\max_K h^K(x_i)$

↳ parameters are already estimated for all cases.

Some observations

- Is there any affect of initial parameter values in performance? (Hyperparameter Tuning/Validation Set)
- ***Is there any affect of different random set of training samples (considering same percentage) in performance of the model?***
- How to check these issues during experimentation?

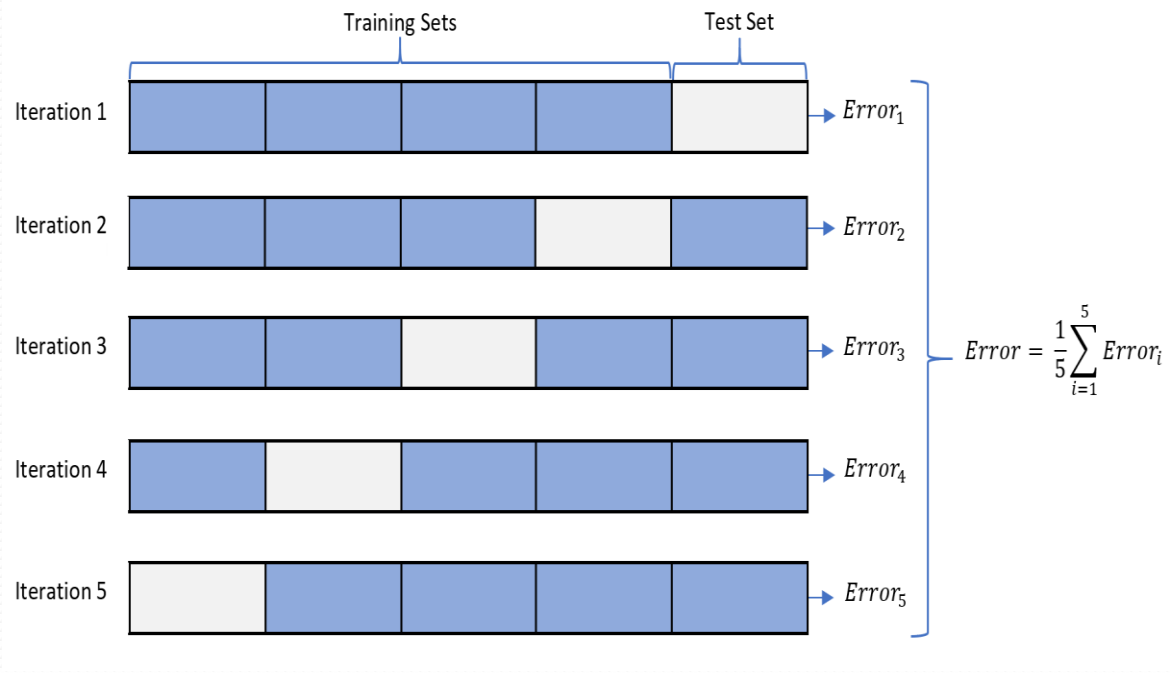
K-fold Cross-Validation

- Cross-validation is a statistical method used to estimate the skill of machine learning models.
- It is a resampling procedure used to evaluate the performance of learning models on different train/test split.
- The model is expected to perform well in generalization when used to make predictions for the samples not used during the training of the model.
- Here, the dataset is divided in train set and test set only.

Procedure

- Shuffle the dataset randomly (Why?)
- Split the dataset into k groups/folds (approximately equal size)
- For each group:
  - Take the group as a hold out or test data set
  - Take the remaining groups as a training data set
  - Fit a model on the training set and evaluate it on the test set
  - Retain the evaluation score and discard the model
- Report the performance score of the model on each group (test set) and average of all the performance score

Validation set??


Training Sets
Test Set
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
$Error_1$
$Error_2$
$Error_3$
$Error_4$
$Error_5$
$Error = \frac{1}{5}\sum_{i=1}^{5} Error_i$

Observation

- Some portion of training set can be used for validation set (for example 10%)
- The value of K: mostly use 5 or 10 for experimentation
- Each pattern is assigned to an individual group and stays in that group for the duration of the procedure
- This means that each sample is given the opportunity to be used in the hold out set/test set 1 (one) time and used to train the model *K-1* times.

Confusion Matrix for Classification

- Confusion Matrix is used to know the performance of a Machine learning classification. It is represented in a matrix form.

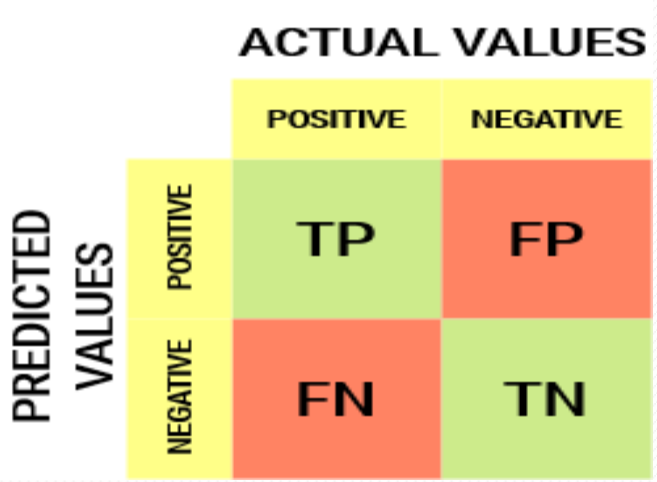
- Confusion Matrix gives a comparison between actual and predicted values.

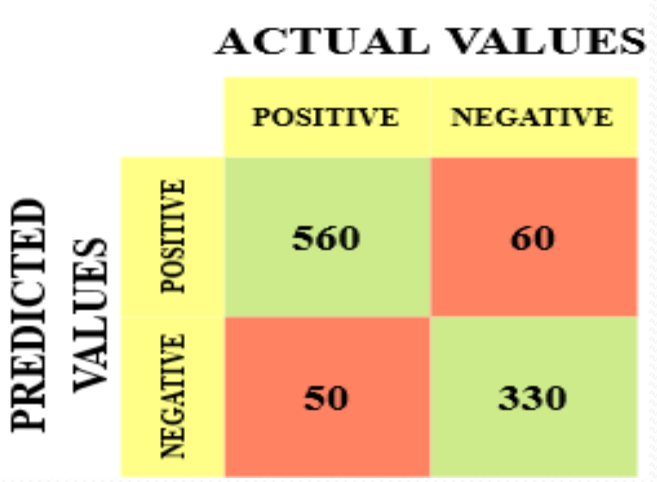
- The confusion matrix is a N x N matrix, where N is the number of classes.

  For 2 class-problem ,we get 2 x 2 confusion matrix.

  For 3 class-problem ,we get 3 X 3 confusion matrix.

Confusion Matrix for Classification

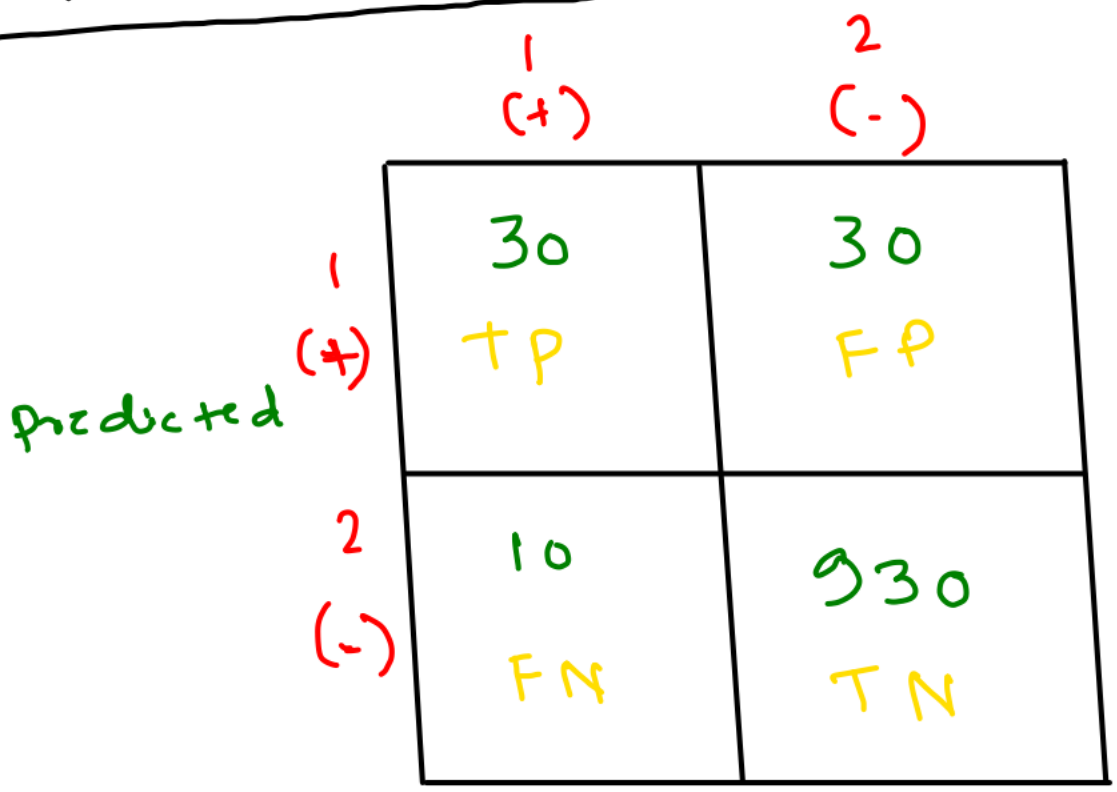




- The target variable has two values: **Positive** or **Negative**
- The **columns** represent the **actual values** of the target variable
- The **rows** represent the **predicted values** of the target variable

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

Confusion matrix



Observation

$$\text{Accuracy} = \frac{30 + 930}{30 + 30 + 10 + 930} = .96$$

good one but not provide the complete idea

$$\text{Accuracy}_{\text{class 1}} = \frac{30}{40} = .75$$

$$\text{Accuracy}_{\text{class 2}} = \frac{930}{960} = .96$$

Suppose, class 1 is more important than class 2

Precision vs. Recall

- **Precision:** Out of all the positive predicted samples how many are truly positive.

$$Precision = \frac{TP}{TP + FP}$$

The precision value lies between 0 and 1.

- **Recall:** How many of the actual positive cases we were able to predict correctly with our model.

$$Recall = \frac{TP}{TP + FN}$$

The recall value lies between 0 and 1.

Precision vs. Recall (Multiclass)

- Calculate precision and recall for each of the classes
- *precision* for the Cat class is the number of correctly predicted Cat photos (4) out of all predicted Cat photos (4+3+6=13), which amounts to 4/13
- the *recall* for Cat is the number of correctly predicted Cat photos (4) out of the number of actual Cat photos (4+1+1=6), which is 4/6

		True/Actual		
		Cat	Fish	Hen
Predicted	Cat	4	6	3
	Fish	1	2	0
	Hen	1	2	6

Some issues related to training samples

- Training samples in one class are significantly fewer than the samples in other classes (class imbalance)

- Sufficient training samples are not available (data scarcity)