BITS Pilani

Pilani Campus

Machine Learning

AIML CZG565

M4 : Linear Models for Classification

Course Faculty of MTech Cluster

BITS – CSIS - WILP

Disclaimer and Acknowledgement

- These content of modules & context under topics are planned by the course owner Dr. Sugata, with grateful acknowledgement to many others who made their course materials freely available online.
- The content for these slides has been obtained from books and various other source on the Internet
- We here by acknowledge all the contributors for their material and inputs.
- We have provided source information wherever necessary

- To ease student's reading , we have added additional slides in this canvas upload, <u>that are not shown in the live class</u> for detailed explanation
- Students are requested to refer to the textbook w.r.t detailed content of this presentation deck shared over canvas

Slide Source / Preparation / Review:

<u>From BITS Pilani WILP</u>**:** Prof.Sugata, Prof.Chetana, Prof.Rajavadhana, Prof.Monali, Prof.Sangeetha, Prof.Swarna, Prof.Pankaj

<u>External:</u> CS109 and CS229 Stanford lecture notes, Dr.Andrew NG and many others who made their course materials freely available online.

Agenda

- Discriminant Functions
- Probabilistic Generative Classifiers
- Probabilistic Discriminative Classifiers
- Logistic Regression
- Applications : Text classification model

Decision Theory
&
Objective of Classification Models

Classification

- Given a collection of records (training set)
  - Each record is by characterized by a tuple (***x***,*y*), where ***x*** is the attribute (feature) set and *y* is the class label
    - ***x*** *aka* attribute, predictor, independent variable, input
    - *Y aka* class, response, dependent variable, output
- Task
  - Learn a model or function that maps each attribute set ***x*** into one of the predefined class labels *y*

Task	Attribute set, ***x***	Class label, *y*
Categorizing email messages	Features extracted from email message header and content	spam or non-spam
Identifying tumor cells	Features extracted from x-rays or MRI scans	malignant or benign cells
Cataloging galaxies	Features extracted from telescope images	Elliptical, spiral, or irregular-shaped galaxies

Logistic Regression Applications

- **Credit Card Fraud** : Predicting if a given credit card transaction is fraud or not
- **Health** : Predicting if a given mass of tissue is benign or malignant
- **Marketing** : Predicting if a given user will buy an insurance product or not
- **Banking** : Predicting if a customer will default on a loan.

Inductive Learning Hypothesis : Interpretation

- Target Concept : **t**
- Discrete : f(x) ∈ {Yes, No, Maybe} Classification
- Continuous : f(x) ∈ [20-100] Regression
- Probability Estimation : f(x) ∈ [0-1]

Sky	**AirTemp**	**Humidity**	**Wind**	**Water**	**Forecast**	***EnjoySport?***
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Decision Theory

- Target Concept : **t**
- Discrete : f(x) ∈ {Yes, No} ie., **t** ∈ {0, 1} Binary Classification
- Continuous : f(x) ∈ [20-100]
- Probability Estimation : f(x) ∈ [0-1]

ML Task : Predict the Employability of interview candidates based on CGPA & IQ

Preprocess Implemented:

Min-Max Normalization on IQ

CGPA	IQ	IQ	Job Offered
5.5	6.7	100	1
5	7	105	0
8	6	90	1
9	7	105	1
6	8	120	0
7.5	7.3	110	0

How does logistic regression handle missing values?

- Replace missing values with column averages (i.e. replace missing values in feature 1 with the average for feature 1).
- Replace missing values with column medians.
- Impute missing values using the other features.
- Remove records that are missing features.
- Use a machine learning technique that uses classification trees, e.g. Decision tree

Decision Theory :

The decision problem: given x, predict t according to a probabilistic model p(x, t)

- Target Concept : **t**
- Discrete : f(x) ∈ {Yes, No} ie., **t** ∈ {0, 1}
- Continuous : f(x) ∈ [20-100]
- Probability Estimation : f(x) ∈ [0-1]

p(x, C_k) is the (central!) inference problem

CGPA	IQ	IQ	Job Offered	P(Job = 1)	
5.5	6.7	100	1	0.8	
X =< 5 ,	7 , >	105	0	0.4 →	= P(C_k | X)
8	6	90	1	0.75	
9	7	105	1	0.95	
6	8	120	0	0.35	
7.5	7.3	110	0	0.4	

Classification Problem: Stages

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}.$$

$$= \frac{p(x, C_k)}{p(x)} = \frac{p(x, C_k)}{\sum_{k=1}^{2} p(x, C_k)}$$

CPGA	IQ	Job - Offered
5.5	6.7	1
5	7	0
8	6	1
9	7	1
6	8	0
7.5	7.3	0

Training Set

Learning algorithm

Induction/ Inference step

Learn Model for $p(x,C_k)$

$p(x,C_{job=1})$ & $p(x,C_{job=0})$

Model

Apply Model to find optimal t

Deduction/ Decision Step

CPGA	IQ	Job - Offered
3	4	?
7	6	?
5.5	8	?

Test Set

Decision Region

Sample Rule / Hypothesis:
IF CGPA>7 Job = 1
Else Job = 0

Training Set

CPGA	IQ	Job - Offered
5.5	6.7	1
5	7	0
8	6	1
9	7	1
6	8	0
7.5	7.3	0

Learning algorithm

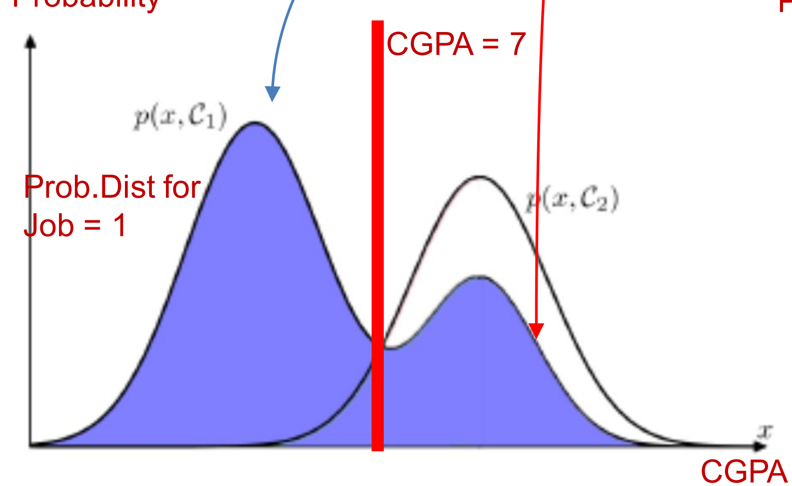
Induction/ Inference step

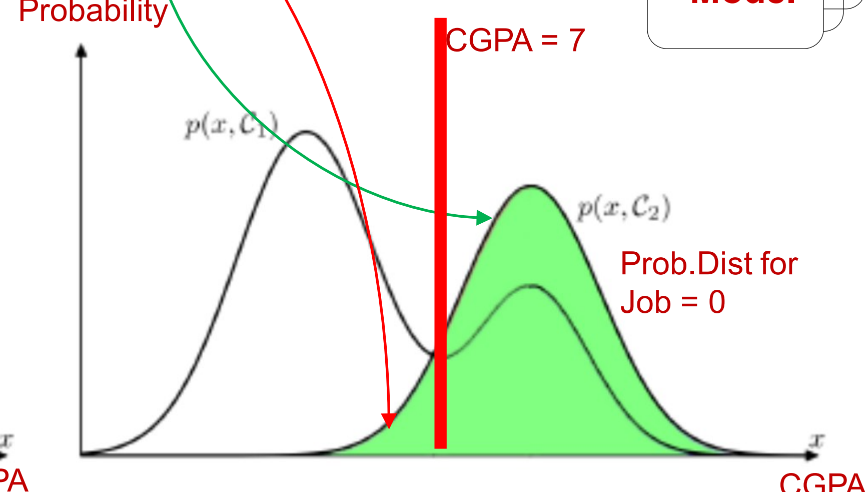
Learn Model for $p(x,C_k)$

Model

Model divides the input space into regions R_k called **decision regions**, one for each class, such that all points in R_k are assigned to class C_k

A mistake occurs when an input vector belonging to class C_1 is assigned to class C_2





Misclassification Rate

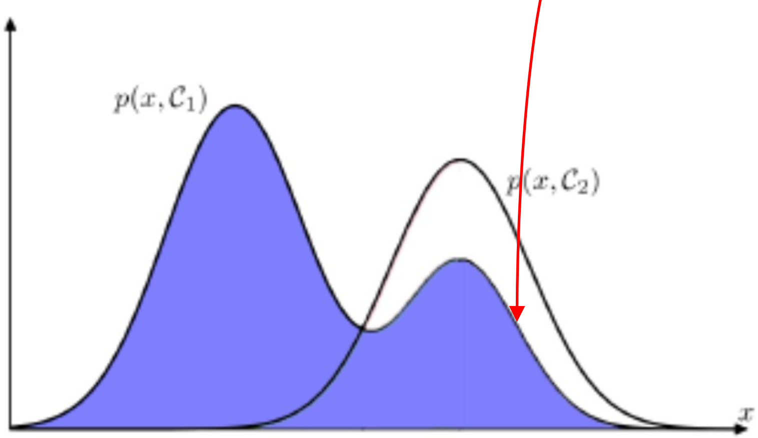
$$p(C_k|x) = \frac{p(x, C_k)}{p(x)}$$

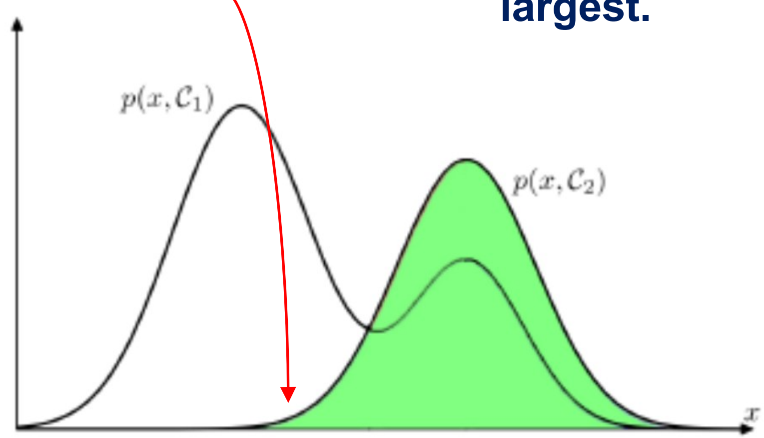
$$\begin{aligned} p(\text{mistake}) &= p(\mathbf{x} \in \mathcal{R}_1, \mathcal{C}_2) + p(\mathbf{x} \in \mathcal{R}_2, \mathcal{C}_1) \\ &= \int_{\mathcal{R}_1} p(\mathbf{x}, \mathcal{C}_2)\,\mathrm{d}\mathbf{x} + \int_{\mathcal{R}_2} p(\mathbf{x}, \mathcal{C}_1)\,\mathrm{d}\mathbf{x}. \end{aligned}$$

To minimize p(mistake), each **x** is assigned to whichever class has the smaller value of the integrand

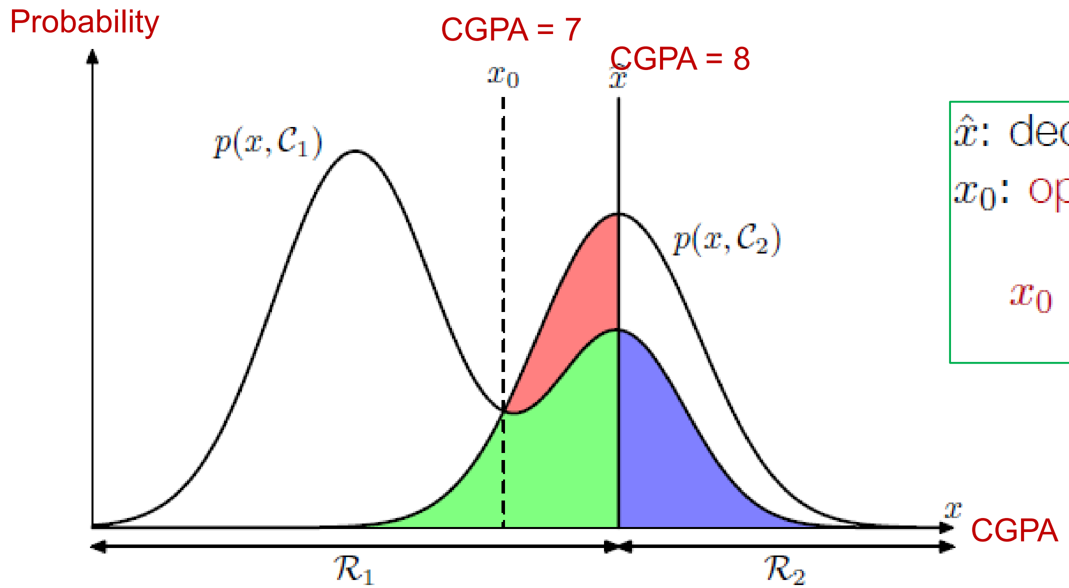
$$\begin{aligned} & p(x, C_1) > p(x, C_2) \\ \Leftrightarrow\ & p(C_1|x)p(x) > p(C_2|x)p(x) \\ \Leftrightarrow\ & p(C_1|x) > p(C_2|x) \end{aligned}$$

The minimum probability of making a mistake is obtained if each value of **x** is assigned to the class for which the **posterior probability $p(C_k|x)$ is largest.**





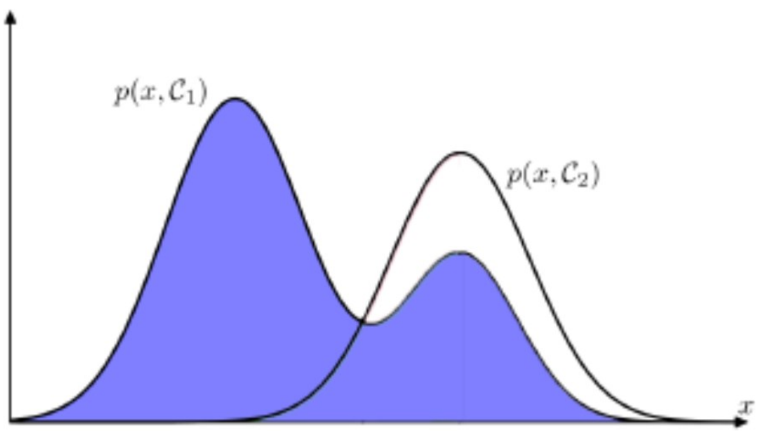
Decision Theory - Summary

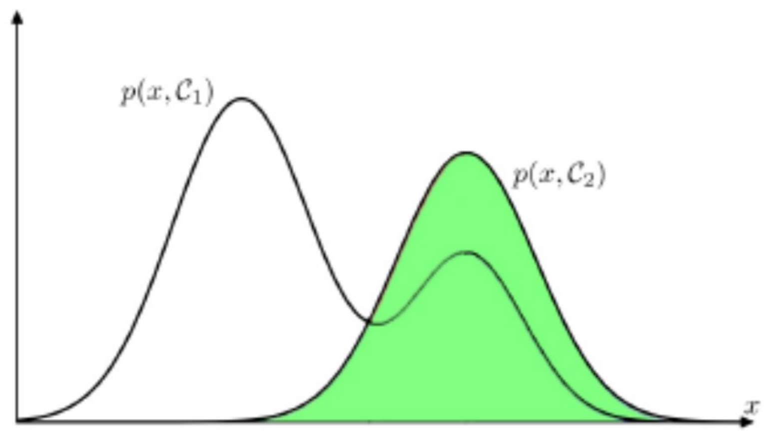


$\hat{x}$: decision boundary.

x_0: optimal decision boundary

$$x_0 : \arg\min_{\mathcal{R}_1}\{p\,(\text{mistake})\}$$





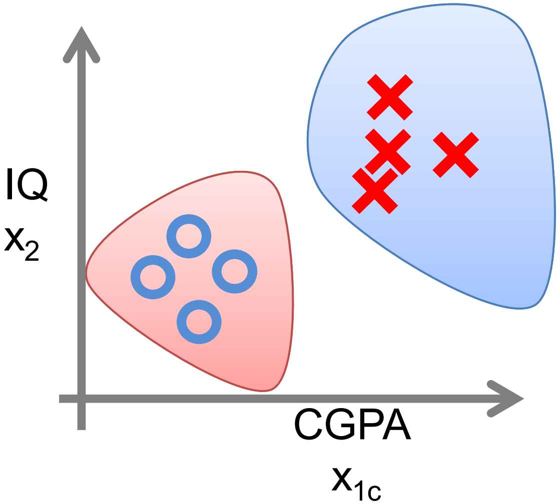
Linear Models for Classification

Types of Classification

Decision Theory: Interpretation

Model Building

Generative



Likelihood

Class Prior Probability

$$P(c \mid x) = \frac{P(x \mid c)\,P(c)}{P(x)}$$

Posterior Probability

Predictor Prior Probability

$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

CPGA	IQ	Job - Offered
5.5	6.7	1
5	7	0
8	6	1
9	7	1
6	8	0
7.5	7.3	0
...	....	.....

$$P(Y \mid X_1 X_2 \ldots X_n) = \frac{P(X_1 X_2 \ldots X_d \mid Y)P(Y)}{P(X_1 X_2 \ldots X_d)}$$

Known as generative models, because by sampling from them it is possible to generate synthetic data points in the input space.

Eg.,Classification: **Naïve Bayes**,

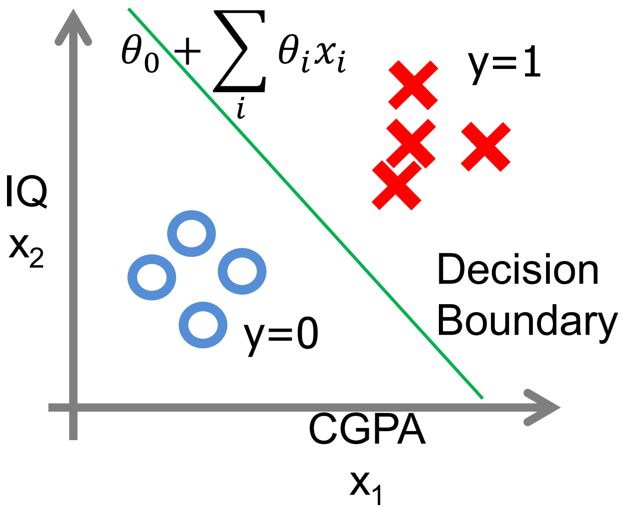
Clustering : **Mixtures of Gaussians**

Types of Classification

Decision Theory: Interpretation

Model Building

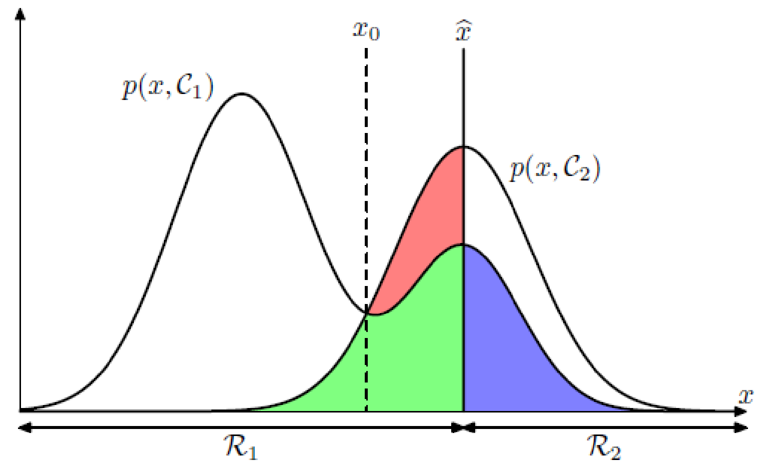
Discriminative



$$\theta_0 + \sum_i \theta_i x_i \geq 0$$

$$\theta_0 + \sum_i \theta_i x_i < 0$$

Logistic regression, SVMs , tree based classifiers (e.g. decision tree) Traditional neural networks, **Nearest neighbor**



$P(c \mid x) =$

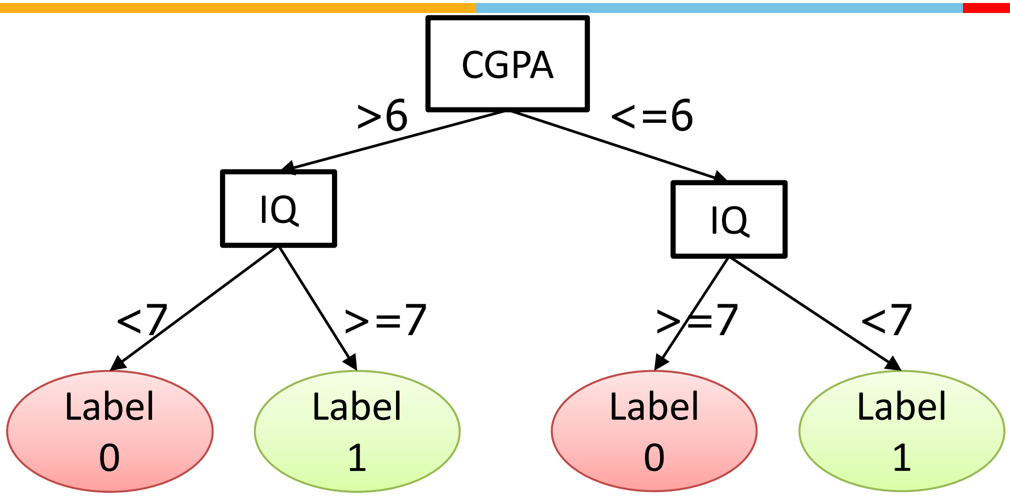
Posterior Probability

CPGA	IQ	Job - Offered
5.5	6.7	1
5	7	0
8	6	1
9	7	1
6	8	0
7.5	7.3	0
…	….	…..

Types of Classification

Decision Theory: Interpretation

Model Building



$IF\ CGPA > 6\ and\ IQ \geq 7$

Job offered = 1

Else If CGPA <=6 and IQ <7

Job offered = 1

Else if CGPA <=6 and IQ >=7

Job offered = 0

………

Logistic regression, SVMs , tree based classifiers (e.g. decision tree) Traditional neural networks, **Nearest neighbor**

CPGA	IQ	Job - Offered
5.5	6.7	1
5	7	0
8	6	1
9	7	1
6	8	0
7.5	7.3	0
…	….	…..

Types of Classification

Generative vs Discriminative Models

- Generative Model
  - Class-conditional probability distribution of attribute/feature set and prior probability of classes are learnt during the training phase
  - Given these learnt probabilities, during inferencing phase, probability of a test record belonging to different classes are calculated and compared.
  - Can result in linear or nonlinear decision surface
- Discriminative Model
  - Given a training set, a function f is learnt that directly maps an attribute/feature vector ***x*** to the output class (y=1 or 0/-1)
  - A linear function f results in linear decision surface

Logistic Regression

Idea :

Given data X and associated binary (0/1) class label Y, Logistic Regression tries to learn a discriminant function P(Y|X)

If Y = 1, P(Y|X) = 1 else P(Y|X) = 0

BITS Pilani

Logistic Regression vs Least Squares Regression

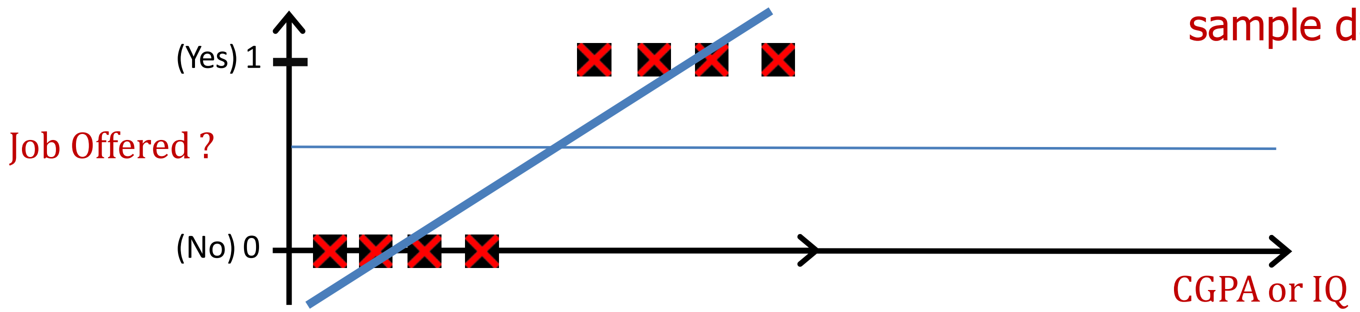


Figure depicts
sample data points

- Independent Attribute : CGPA or IQ
- Can we solve the problem using linear regression? E.g., fit a straight line and define a threshold at 0.5
- Threshold classifier output $h_\theta(x)$ at 0.5:

$$h_\theta(x) = \text{estimated probability that } y = 1 \text{ on input } x$$

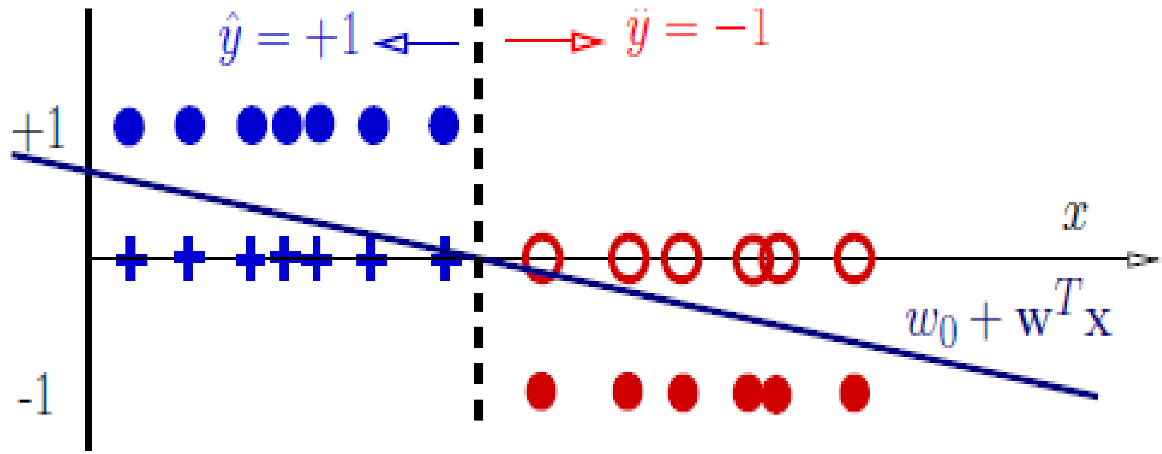
A Discriminant function f (x)
directly map input to class labels
In two-class problem, f (.) is binary valued

If $h_\theta(x)$ >=0.5, predict "y = 1"

If $h_\theta(x)$<0.5 , predict "y = 0"

In this use case : $h_\theta(x)$ = 0.7, implies that there is 70% of chance of the candidate being selected in the interview

Decision Rules



- Classifier:

  $f(\mathbf{x}, \mathbf{w}) = w_o + \mathbf{w}^T \mathbf{x}$ (linear discriminant function)

- Decision rule is

$$y = \begin{cases} 1 & \text{if } f(\mathbf{x}, \mathbf{w}) \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

- Mathematically

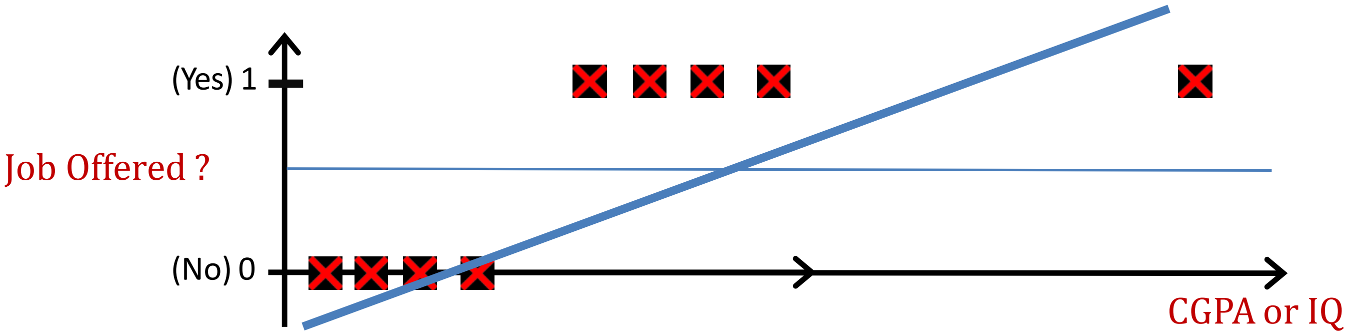
  $y = \text{sign}(w_0 + \mathbf{w}^T \mathbf{x})$

- This specifies a linear classifier: it has a linear boundary (hyperplane)

  $w_0 + \mathbf{w}^T \mathbf{x} = 0$

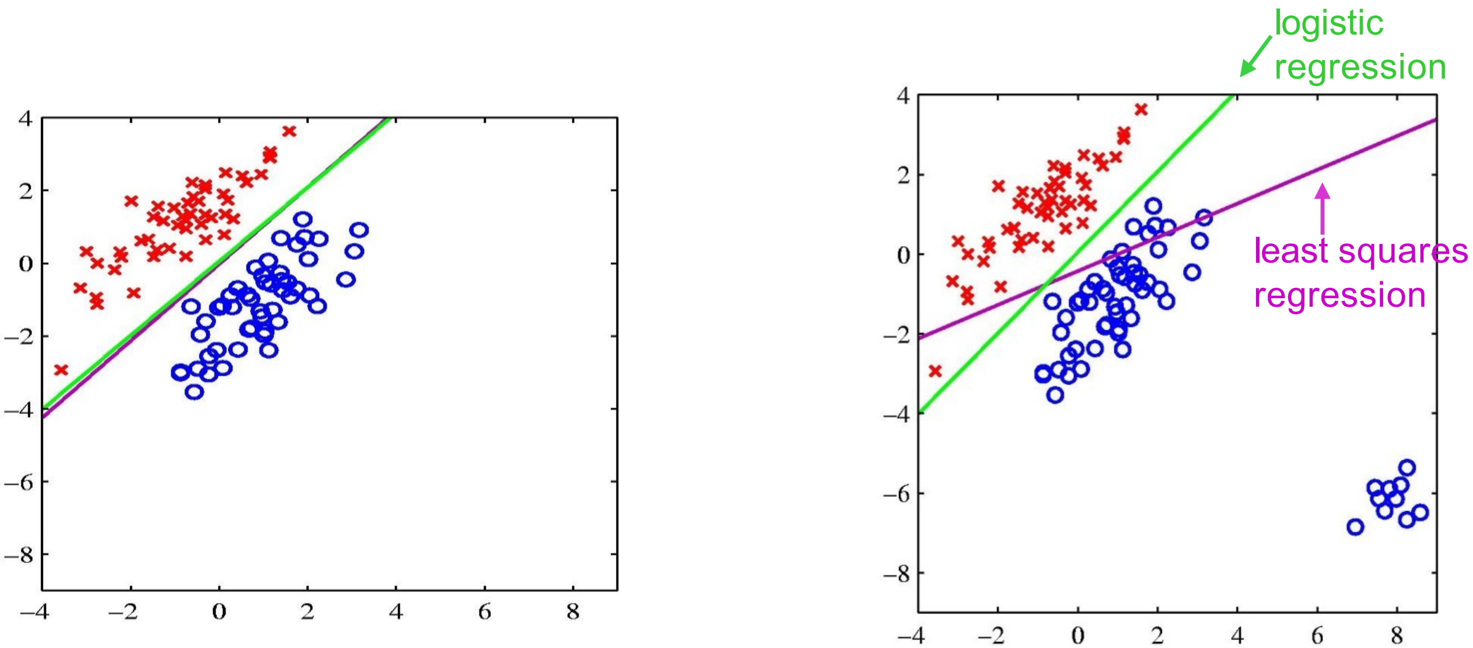
A discriminant is a function that takes an input vector x and assigns it to one of K classes, denoted C_k.

Logistic Regression vs Least Squares Regression



Failure due to adding a new point

Logistic Regression vs Least Squares Regression



The right-hand plot shows the corresponding results obtained when extra data points are added at the bottom left of the diagram, showing that least squares is highly sensitive to outliers, unlike logistic regression.

Logistic Regression vs Least Squares Regression

- **Linear Regression** could help us predict the student's test score on a scale of 0 - 100. Linear regression predictions are continuous (numbers in a range).
- **Logistic Regression** could help use predict whether the student passed or failed. Logistic regression predictions are discrete (only specific values or categories are allowed). We can also view probability scores underlying the model's classifications.

Intuition behind the model:

Classification requires discrete values: y = 0 or 1

For linear Regression output values: $h_\theta(x)$ can be much > 1 or much < 0

Logistic Regression: $0 \leq h_\theta(x) \leq 1$

Sigmoid Function

- Sigmoid/logistic function takes a real value as input and outputs another value between 0 and 1
- That framework is called logistic regression
  - Logistic: A special mathematical sigmoid function it uses
  - Regression: Combines a weight vector with observations to create an answer

- Want $0 \leq h_\theta(x) \leq 1$
- $h_\theta(x) = g(\theta^\top x)$,

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

- Sigmoid function
- Logistic function

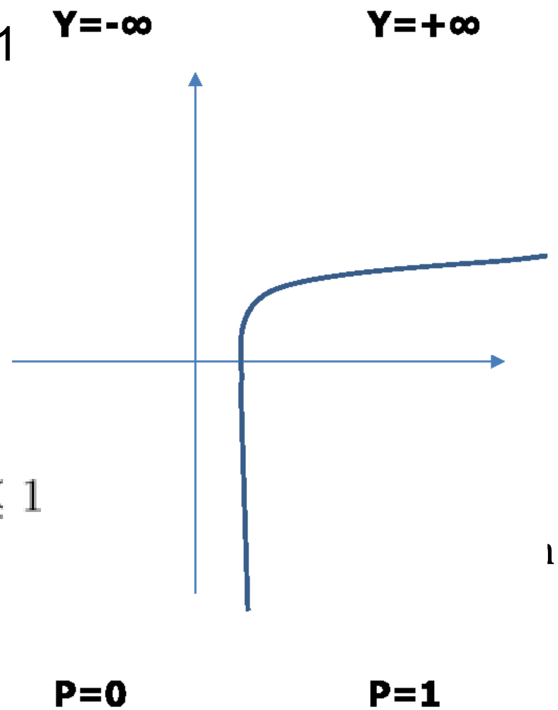
Logistic Regression vs Least Squares Regression

Classification: $y = 0$ or 1

$h_\theta(x)$ can be > 1 or < 0

What we need: $0 \le h_\theta(x) \le 1$

Logistic Regression:



$$Y = \theta_0 + \sum_i \theta_i x_i$$

$$p = \frac{1}{1+e^{-y}} \qquad h_\theta(x) = g(\theta^T x)$$

$$\frac{p}{1-p} = \theta_0 + \sum_i \theta_i x_i$$

Classification – Linear Vs Non Linear Decision Boundary

- At decision boundary output of logistic regression is 0.5
- Classes are separated by a linear decision surface (e.g., straight line in 2-dimensional feature/attribute space)
  - If for a given record, linear combination of features x_i is >= 0, i.e.,

$$w_0 + \sum_i w_i x_i \geq 0$$

  it belongs to one class (say, $y = 1$), else it belongs to the other class (say, y=0 or -1)
  - w_is are learned during the training (induction) phase of the classifier.
  - Learnt w_is are applied to a test record during the deduction / inferencing phase.
- In nonlinear classification, classes are separated by a non-linear surface

Logistic Regression – Sample Linear Boundary

- At decision boundary output of logistic regression is 0.5
- $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$
  - e.g., $\theta_0 = -3,\ \theta_1 = 1,\ \theta_2 = 1$

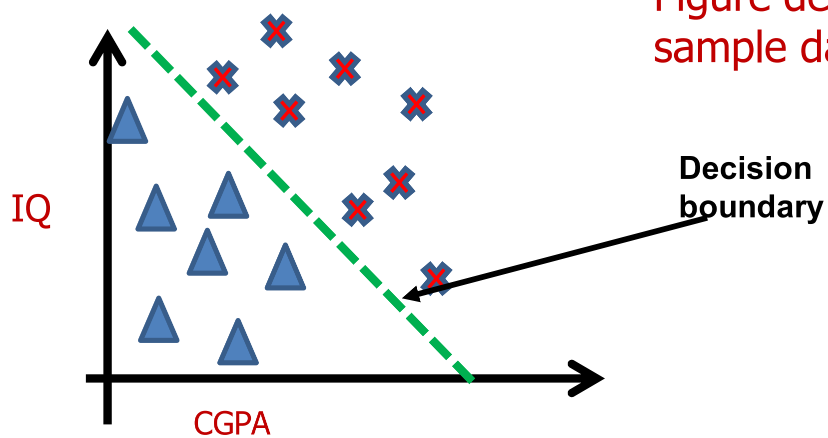
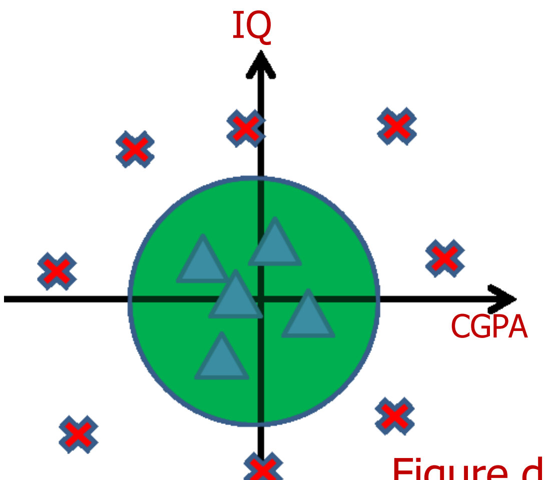


Figure depicts sample data points

- Predict "$y = 1$" if $-3 + x_1 + x_2 \geq 0$

Slide credit: Andrew Ng

Logistic Regression – Sample Non-Linear Boundary

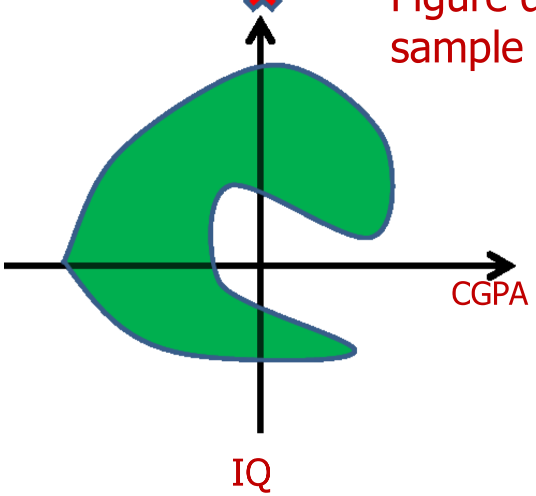


- $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$

E.g., $\theta_0 = -1, \theta_1 = 0, \theta_2 = 0, \theta_3 = 1, \theta_4 = 1$

- Predict "$y = 1$" if $-1 + x_1^2 + x_2^2 \geq 0$

Figure depicts sample data points



- $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^2 x_2^2 + \theta_6 x_1^3 x_2 + \cdots)$

Slide credit: Andrew Ng

Learning model parameters

- Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$
- m examples

$$x \in \begin{bmatrix} x_0 \\ x_1 \\ \cdots \\ x_n \end{bmatrix} \qquad x_0 = 1, y \in \{0, 1\}$$

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

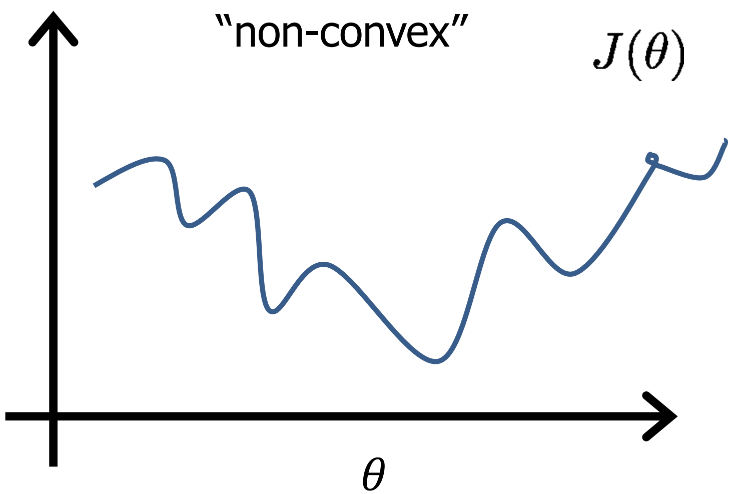
- **How to choose parameters (feature weights) ?**

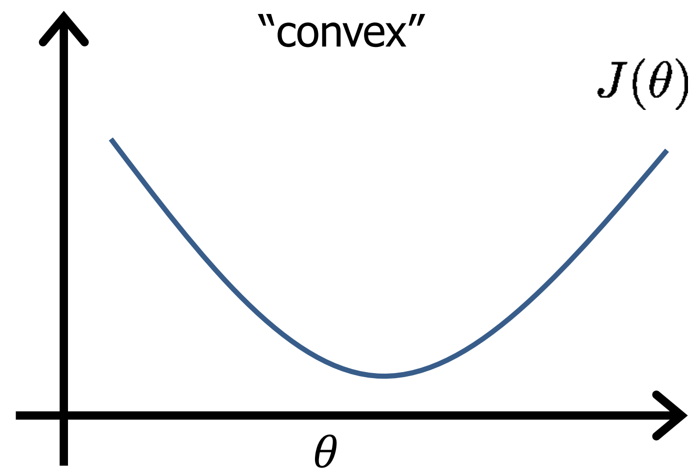
Notion of Cost Function in Classification

BITS Pilani

Logistic Regression

- Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$
- How to choose parameters (feature weights)?





Error (Cost) Function

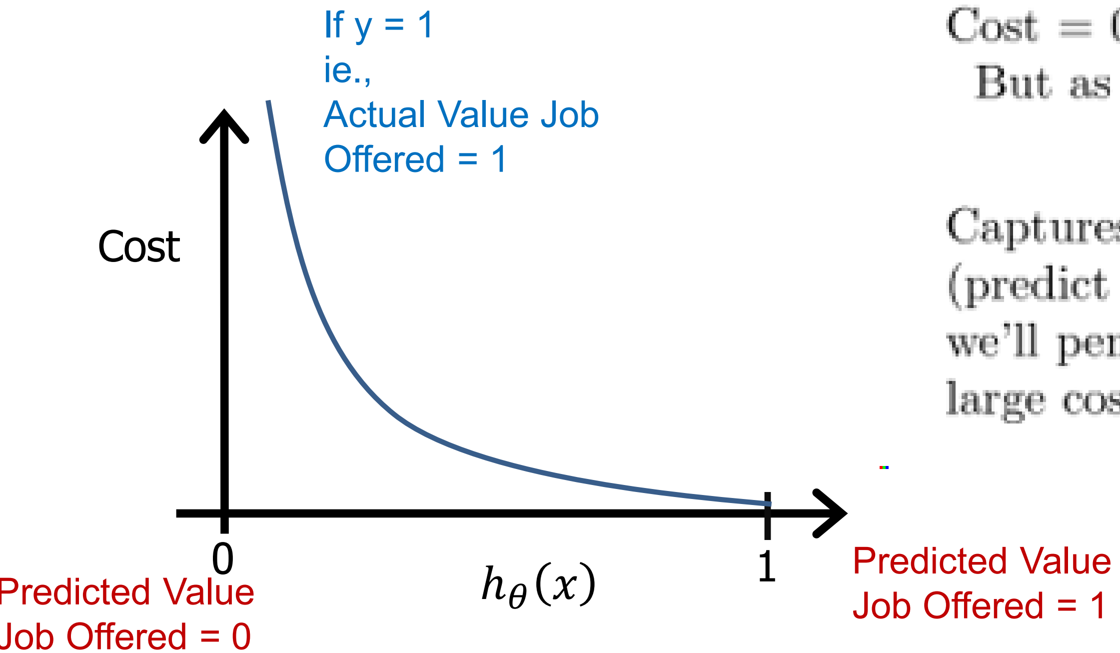
- Our prediction function is non-linear (due to sigmoid transform)
- Squaring this prediction as we do in MSE results in a non-convex function with many local minima.
- If our cost function has many local minimas, gradient descent may not find the optimal global minimum.
- So instead of Mean Squared Error, we use a error/ cost function called Cross-Entropy, also known as Log Loss.

Cross Entropy

- Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1.
- Cross-entropy loss increases as the predicted probability diverges from the actual label. So predicting a probability of .012 when the actual observation label is 1 would be bad and result in a high loss value.
- A perfect model would have a log loss of 0.
- Cross-entropy loss can be divided into two separate cost functions: one for y=1 and one for y=0.

Logistic regression cost function (cross entropy)

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



$\mathrm{Cost} = 0$ if $y = 1, h_\theta(x) = 1$

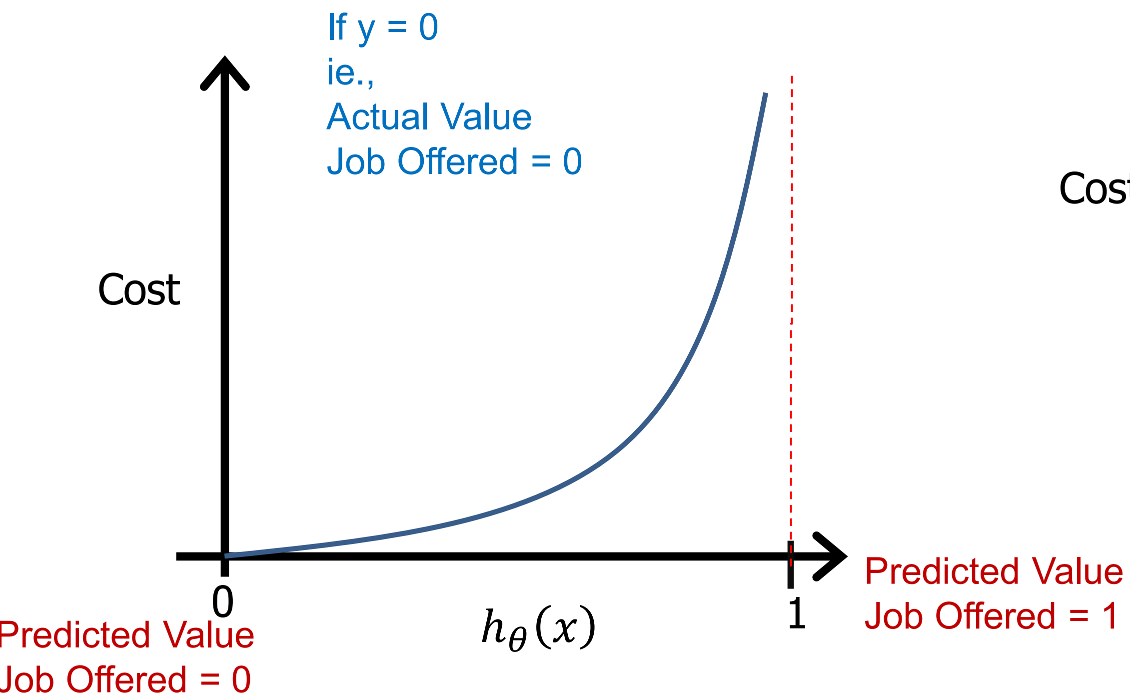
But as $h_\theta(x) \to 0$

$Cost \to \infty$

Captures intuition that if $h_\theta(x) = 0$, (predict $P(y = 1|x;\theta) = 0$), but $y = 1$, we'll penalize learning algorithm by a very large cost.

Logistic regression cost function

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



Cost=0; If y=0 and $h_\theta(x)$=0

Cost function

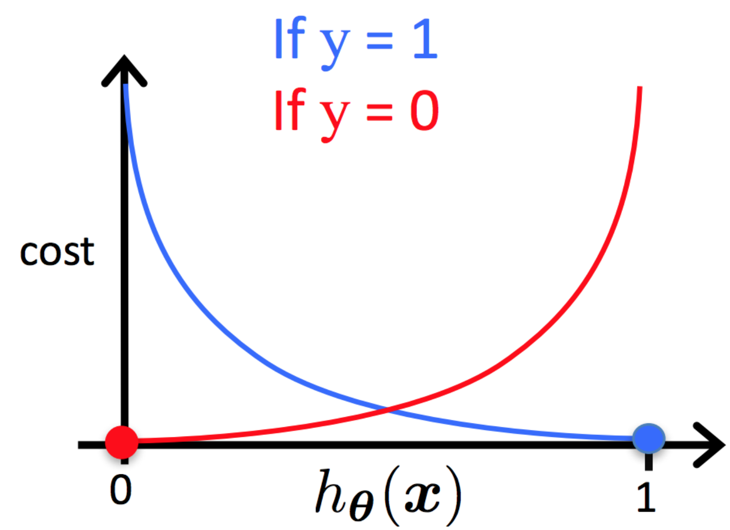
$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

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

To fit parameters θ : Apply Gradient Descent Algorithm $\min_\theta J(\theta)$

To make a prediction given new :

Output : $h_\theta(x) = \dfrac{1}{1 + e^{-\theta^\top x}}$



Gradient Descent Algorithm

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

Goal: $\min_\theta J(\theta)$

Repeat

{

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

$$\boxed{\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}}$$

Learning: fit parameter θ

$\min_\theta J(\theta)$

Prediction: given new x

Output $h_\theta(x) = \frac{1}{1+e^{-\theta^{\mathsf{T}} x}}$

Gradient Descent Algorithm

Linear Regression

Repeat {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

}

$$\boxed{h_\theta(x) = \theta^\top x}$$

Logistic Regression

Repeat {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

}

$$\boxed{h_\theta(x) = \frac{1}{1 + e^{-\theta^\top x}}}$$

$$\boxed{h_\theta(x) = \frac{1}{1 + e^{-(\theta_0 + \theta_1 CGPA + \theta_2 IQ)}}}$$

Slide credit: Andrew Ng

Logistic regression more generally

- Logistic regression **when Y is not Boolean (but still discrete-valued).**
- Now $y \in \{y_1 \ldots y_R\}$: learn R-1 sets of weights

This is equivalent to sigmoid function. Multiply numerator & denominator by exp($-\theta^T x$) in the original form to get this

For *k<R*
(all the 1st ,2nd ,….,(R-1)th label)

$$P(Y = y_k|X) = \frac{\exp(w_{k0} + \sum_{i=1}^{n} w_{ki}X_i)}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)}$$

For *k=R*

$$P(Y = y_R|X) = \frac{1}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)}$$

Eg., If the class has three distinct values with assumption that the dataset has 6 features and linear decision boundary works:

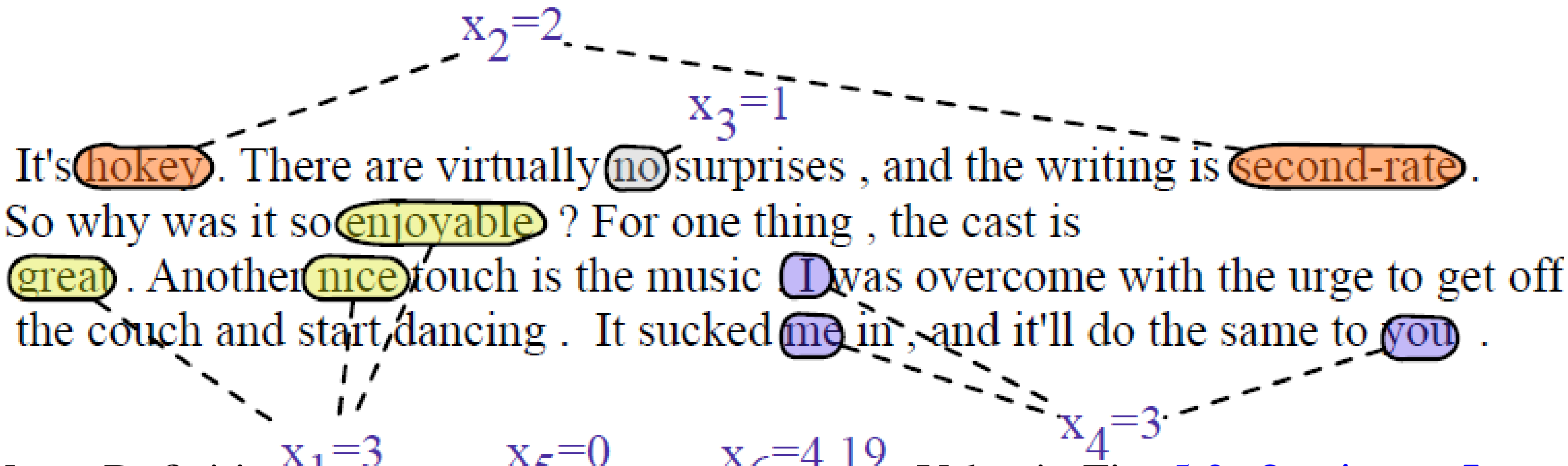
Classifier learns two set of weights. Each set of weight has $\{\theta_0, \theta_1, \theta_2 \ldots\ldots \theta_6\}$

For the third class value(label: k=Rth value) it uses the second formula for estimation.

Application of Logistic Regression
&
Problem Types

BITS Pilani

Example: Sentiment Analysis – With Engineered features



Sentiment Features

Var	Definition	Value in Fig. 5.2
x_1	count(positive lexicon) $\in$ doc)	3
x_2	count(negative lexicon) $\in$ doc)	2
x_3	$\begin{cases} 1 & \text{if “no”} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	1
x_4	count(1st and 2nd pronouns $\in$ doc)	3
x_5	$\begin{cases} 1 & \text{if “!”} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	0
x_6	log(word count of doc)	$\ln(66) = 4.19$

Classifying sentiment using logistic regression

$$\text{Suppose w} = [2.5, -5.0, -1.2, 0.5, 2.0, 0.7]$$

$$\text{b} = 0.1$$

$$\begin{aligned} p(+|x) = P(Y=1|x) &= \sigma(w \cdot x + b) \\ &= \sigma([2.5, -5.0, -1.2, 0.5, 2.0, 0.7] \cdot [3, 2, 1, 3, 0, 4.19] + 0.1) \\ &= \sigma(.833) \\ &= 0.70 \end{aligned}$$

$$\begin{aligned} p(-|x) = P(Y=0|x) &= 1 - \sigma(w \cdot x + b) \\ &= 0.30 \end{aligned}$$

Logistic Regression – Fit a Model – Apply Gradient Descent

CGPA	IQ	IQ	Job Offered
5.5	6.7	100	1
5	7	105	0
8	6	90	1
9	7	105	1
6	8	120	0
7.5	7.3	110	0

$$\theta_0 := \theta_0 - 0.3\frac{1}{6}\sum_{i=1}^{6}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)(1)$$

$$\theta_{CGPA} := \theta_{CGPA} - 0.3\frac{1}{6}\sum_{i=1}^{6}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_{CGPA}^{(i)}$$

$$\theta_{IQ} := \theta_{IQ} - 0.3\frac{1}{6}\sum_{i=1}^{6}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_{IQ}^{(i)}$$

Hyper parameters:
Learning Rate = 0.3
Initial Weights = (0.5, 0.5,0.5)
Regularization Constant = 0

$\theta^T X$ = 0.5 +0.5 CGPA + 0.5 IQ

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

$$h_\theta(x) = \frac{1}{1 + e^{-(0.5 + 0.5\,CGPA + 0.5\,IQ)}}$$

Approx. : New weights

θ_0 = 0.4

$\theta_{1=CGPA}$ =-0.4

$\theta_{2=IQ} = -0.6$

Logistic Regression – Inference & Interpretation

CGPA	IQ	IQ	Job Offered
5.5	6.7	100	1
5	7	105	0
8	6	90	1
9	7	105	1
6	8	120	0
7.5	7.3	110	0

Assume : 0.4+0.3CGPA-0.45IQ

Predict the Job offered for a candidate : (5, 6)
h(x) = 0.31
Y-Predicted = 0 / No

Note :
The exponential function of the regression coefficient ($e^{w\text{-}cpga}$) is the odds ratio associated with a one-unit increase in the cgpa.
+ The odd of being offered with job increase by a factor of 1.35 for every unit increase in the CGPA [np.exp(model.params)]

Regularization

Note : This topic is already covered in the module 3 and the implementation remains the same. **Only one points added here w.r.t interpretation for logistic regression**

BITS Pilani

Ways to Control Overfitting – Interpretation of Hyper parameter

- Regularization

$$Loss(S) = \sum_{i}^{n} Loss(\hat{y_i}, y_i) \quad + \alpha \sum_{j}^{\#\,Weights} |\theta_j|$$

Note:
The hyperparameter controlling the regularization strength of a Scikit-Learn LogisticRegression model is not alpha (as in other linear models), but its inverse: C. The higher the value of C, the less the model is regularized.

Evaluation of Classifiers

Using Another Example

Following contents are common for all the classifiers

Classifier Evaluation Metrics: Confusion Matrix

Confusion Matrix:

Given *m* classes, an entry, ***$CM_{i,j}$*** in a **confusion matrix** indicates # of tuples in class *i* that were labeled by the classifier as class *j*

May have extra rows/columns to provide totals

- **True Positive (TP):** It refers to the number of predictions where the classifier correctly predicts the positive class as positive.
- **True Negative (TN):** It refers to the number of predictions where the classifier correctly predicts the negative class as negative.
- **False Positive (FP):** It refers to the number of predictions where the classifier incorrectly predicts the negative class as positive.
- **False Negative (FN):** It refers to the number of predictions where the classifier incorrectly predicts the positive class as negative.

Predicted class ->	C_1	$\neg C_1$
Actual class⇓		
C_1	**True Positives (TP)**	**False Negatives (FN)**
$\neg C_1$	**False Positives (FP)**	**True Negatives (TN)**

Classifier Evaluation Metrics: Confusion Matrix

Confusion Matrix:

Classifier Accuracy, or recognition rate:
percentage of test set tuples that are correctly classified

Accuracy = (TP + TN)/All

most effective when the class distribution is relatively balanced

Classification Error/ Misclassification rate:
1 – accuracy, or
= (FP + FN)/All

Predicted class ->	C_1	$\neg C_1$
Actual class⇓		
C_1	**True Positives (TP)**	**False Negatives (FN)**
$\neg C_1$	**False Positives (FP)**	**True Negatives (TN)**

Evaluation of Classification Model

Confusion Matrix

Mileage (in kmpl)	*Car Price (in cr)*
9.8	High
9.12	Low
9.5	High
10	Low
….	…

	PREDICTED CLASS	
ACTUAL CLASS	Class= Low	Class= High
Class= Low	a (TP)	b (FN)
Class= High	c (FP)	d (TN)

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Unseen Data	
Mileage (in kmpl)	***Car Price (in cr)***
7.5	High
10	Low
….	…..

Most effective when the class distribution is relatively balanced

Model 1

$$\text{CarPrice} = \frac{1}{1+e^{-8.5 + 0.5\text{ Mileage} - 1.5\text{ Mileage}^2}}$$

Accuracy: 99%

Model 2

$$\text{CarPrice} = \frac{1}{1+e^{5.5-1.5\text{ Mileage}}}$$

Accuracy: 50%

Evaluation of Classification Model

Confusion Matrix

	PREDICTED CLASS		
ACTUAL CLASS		Class=Low	Class=High
	Class=Low	0 (TP)	10 (FN)
	Class=High	0 (FP)	990 (TN)

If a model predicts everything to be class NO, accuracy is 990/1000 = 99 %. This is misleading because this trivial model does not detect any class YES example Detecting the rare class is usually more interesting (e.g., frauds, intrusions, defects, etc)

	PREDICTED CLASS		
ACTUAL CLASS		Class=Low	Class=High
	Class=Low	10 (TP)	0 (FN)
	Class=High	500 (FP)	490 (TN)

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Which model is better?

Model 1

$$\text{CarPrice} = \frac{1}{1+e^{-8.5 + 0.5\ \text{Mileage} - 1.5\ \text{Mileage}^2}}$$

Accuracy: 99%

Model 2

$$\text{CarPrice} = \frac{1}{1+e^{5.5-1.5\ \text{Mileage}}}$$

Accuracy: 50%

Evaluation of Classification Model

Confusion Matrix

	PREDICTED CLASS		
ACTUAL CLASS		Class= Low	Class= High
	Class= Low	a (TP)	b (FN)
	Class= High	c (FP)	d (TN)

The F-score (also known as the F1 score or F-measure, combines precision and recall into a single score . F1-score is a better metric when there are imbalanced classes (More on this in upcoming slides)

F-score
= 2 * (precision * recall) / (precision + recall)

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

$$ErrorRate = 1 - accuracy$$

$$Precision = Positive\ Predictive\ Value = \frac{TP}{TP + FP}$$

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

$$Specificity = TN\ Rate = \frac{TN}{TN + FP}$$

$$FP\ Rate = \alpha = \frac{FP}{TN + FP} = 1 - specificity$$

$$FN\ Rate = \beta = \frac{FN}{FN + TP} = 1 - sensitivity$$

$$Power = sensitivity = 1 - \beta$$

Evaluation of Classifiers

Given below is a confusion matrix for medical data where the class values are yes and no for a class label attribute, cancer. Calculate the accuracy of the classifier.

Classes	*yes*	*no*	*Total*	*Recognition (%)*
yes	**90**	**210**	300	30.00
no	**140**	**9560**	9700	98.56
Total	230	9770	10,000	96.40

Confusion matrix for the classes *cancer* = *yes* and *cancer* = *no*.

Actual Class\Predicted class	cancer = yes	cancer = no	Total	Recognition(%)
cancer = yes	**90**	**210**	300	30.00 (*sensitivity*
cancer = no	**140**	**9560**	9700	98.56 (*specificity*)
Total	230	9770	10000	96.40 (*accuracy*)

Which Classifier is better?

Low Skew Case

T1	PREDICTED CLASS	
ACTUAL CLASS	Class=Yes	Class=No
Class=Yes	50	50
Class=No	1	99

$$\text{Precision (p)} = 0.98$$
$$\text{TPR} = \text{Recall (r)} = 0.5$$
$$\text{FPR} = 0.01$$
$$\text{TPR/FPR} = 50$$
$$\text{F-measure} = 0.66$$

T2	PREDICTED CLASS	
ACTUAL CLASS	Class=Yes	Class=No
Class=Yes	99	1
Class=No	10	90

$$\text{Precision (p)} = 0.9$$
$$\text{TPR} = \text{Recall (r)} = 0.99$$
$$\text{FPR} = 0.1$$
$$\text{TPR/FPR} = 9.9$$
$$\text{F-measure} = 0.94$$

T3	PREDICTED CLASS	
ACTUAL CLASS	Class=Yes	Class=No
Class=Yes	99	1
Class=No	1	99

$$\text{Precision (p)} = 0.99$$
$$\text{TPR} = \text{Recall (r)} = 0.99$$
$$\text{FPR} = 0.01$$
$$\text{TPR/FPR} = 99$$
$$\text{F-measure} = 0.99$$

Which Classifier is better?

Medium Skew Case

T1	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	50	50
	Class=No	10	990

$$\text{Precision (p)} = 0.83$$
$$\text{TPR} = \text{Recall (r)} = 0.5$$
$$\text{FPR} = 0.01$$
$$\text{TPR/FPR} = 50$$
$$\text{F} - \text{measure} = 0.62$$

T2	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	99	1
	Class=No	100	900

$$\text{Precision (p)} = 0.5$$
$$\text{TPR} = \text{Recall (r)} = 0.99$$
$$\text{FPR} = 0.1$$
$$\text{TPR/FPR} = 9.9$$
$$\text{F} - \text{measure} = 0.66$$

T3	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	99	1
	Class=No	10	990

$$\text{Precision (p)} = 0.9$$
$$\text{TPR} = \text{Recall (r)} = 0.99$$
$$\text{FPR} = 0.01$$
$$\text{TPR/FPR} = 99$$
$$\text{F} - \text{measure} = 0.94$$

Which Classifier is better?

High Skew Case

T1	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	50	50
	Class=No	100	9900

$$\text{Precision (p)} = 0.3$$
$$\text{TPR} = \text{Recall (r)} = 0.5$$
$$\text{FPR} = 0.01$$
$$\text{TPR/FPR} = 50$$
$$\text{F} - \text{measure} = 0.375$$

T2	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	99	1
	Class=No	1000	9000

$$\text{Precision (p)} = 0.09$$
$$\text{TPR} = \text{Recall (r)} = 0.99$$
$$\text{FPR} = 0.1$$
$$\text{TPR/FPR} = 9.9$$
$$\text{F} - \text{measure} = 0.165$$

T3	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	99	1
	Class=No	100	9900

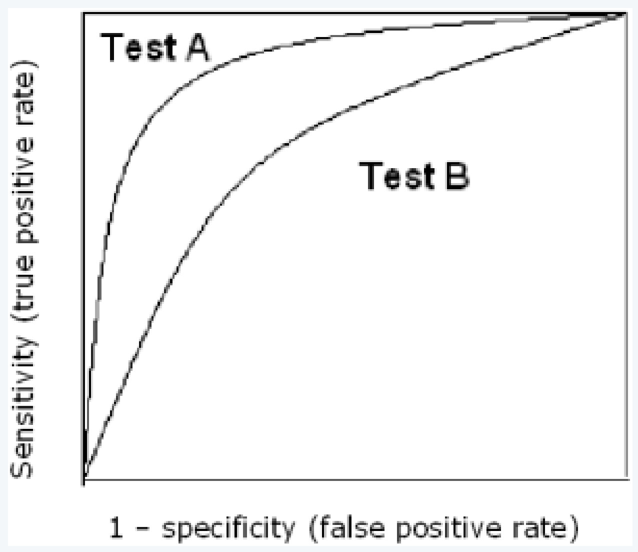
$$\text{Precision (p)} = 0.5$$
$$\text{TPR} = \text{Recall (r)} = 0.99$$
$$\text{FPR} = 0.01$$
$$\text{TPR/FPR} = 99$$
$$\text{F} - \text{measure} = 0.66$$

Which Model should you use?

	False Positive Rate	False Negative Rate
Model 1	41%	3%
Model 2	5%	25%

Mistakes have different costs:

- Disease Screening – LOW FN Rate
- Spam filtering – LOW FP Rate

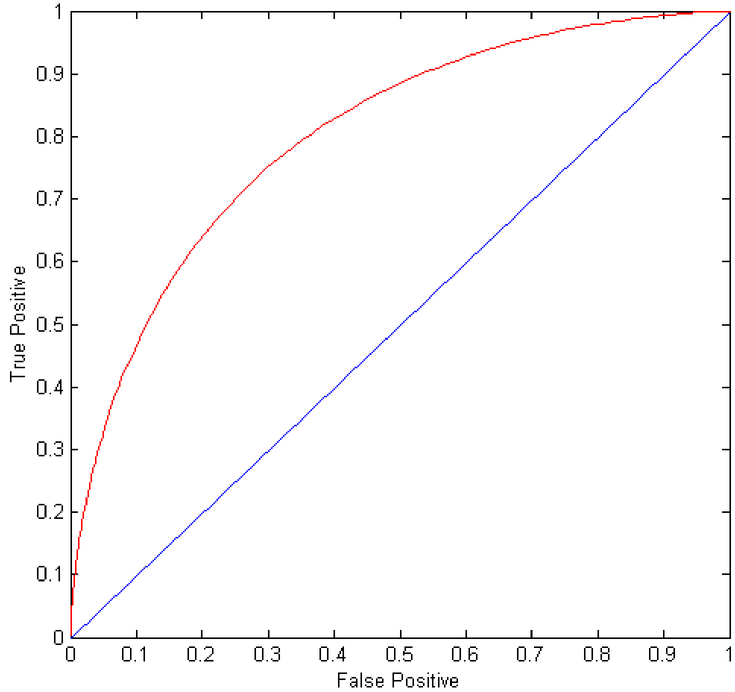


Conservative vs Aggressive settings:

- The same application might need multiple tradeoffs

ROC (Receiver Operating Characteristic)

- A graphical approach for displaying trade-off between detection rate and false alarm rate
- AUC represents degree or measure of separability. It tells how much model is capable of distinguishing between classes.
- Developed in 1950s for signal detection theory to analyze noisy signals
- ROC curve plots TPR against FPR
  - Performance of a model represented as a point in an ROC curve
- Usage
  - Threshold selection
  - Performance assessment
  - Classifier comparison

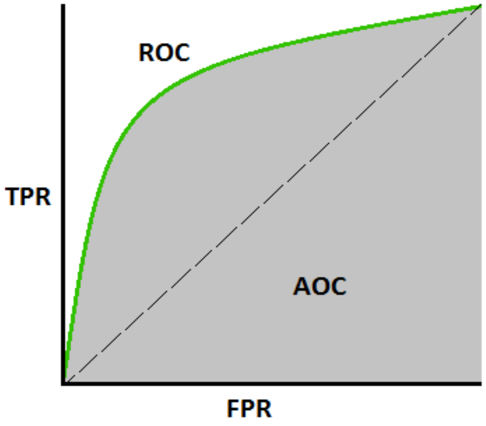


(TPR,FPR):

- (0,0): declare everything to be negative class
- (1,1): declare everything to be positive class
- (1,0): ideal
- Diagonal line:
  - Random guessing
  - Below diagonal line:
    - prediction is opposite of the true class

ROC (Receiver Operating Characteristic)

- To draw ROC curve, classifier must produce continuous-valued output
  - Outputs are used to rank test records, from the most likely positive class record to the least likely positive class record
  - By using different thresholds on this value, we can create different variations of the classifier with TPR/FPR tradeoffs
- Many classifiers produce only discrete outputs (i.e., predicted class)
  - How to get continuous-valued outputs?
    - Decision trees, rule-based classifiers, neural networks, Bayesian classifiers, k-nearest neighbors, SVM



innovate achieve lead

How to Construct an ROC curve

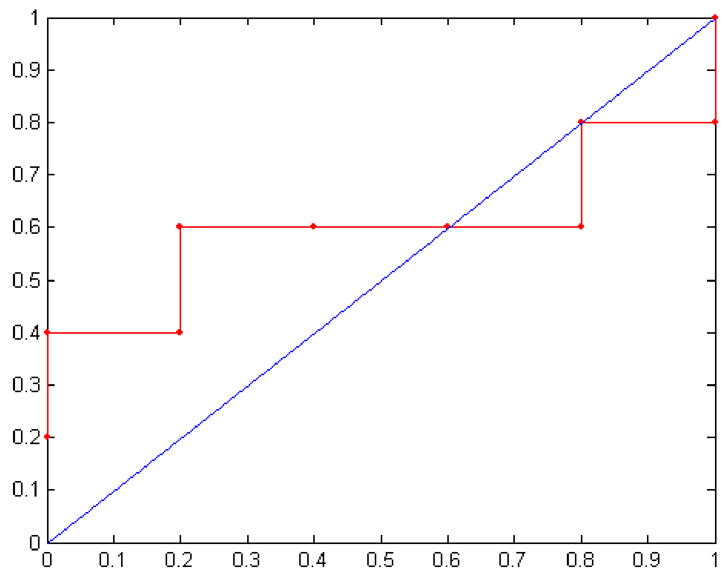
Instance	Score	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

1. Use a classifier that produces a continuous-valued score for each instance
   – The more likely it is for the instance to be in the + class, the higher the score
2. Sort the instances in decreasing order according to the score
3. Apply a threshold at each unique value of the score
4. Count the number of TP, FP, TN, FN at each threshold
   - TPR = TP/(TP+FN)
   - FPR = FP/(FP + TN)

How to construct an ROC curve

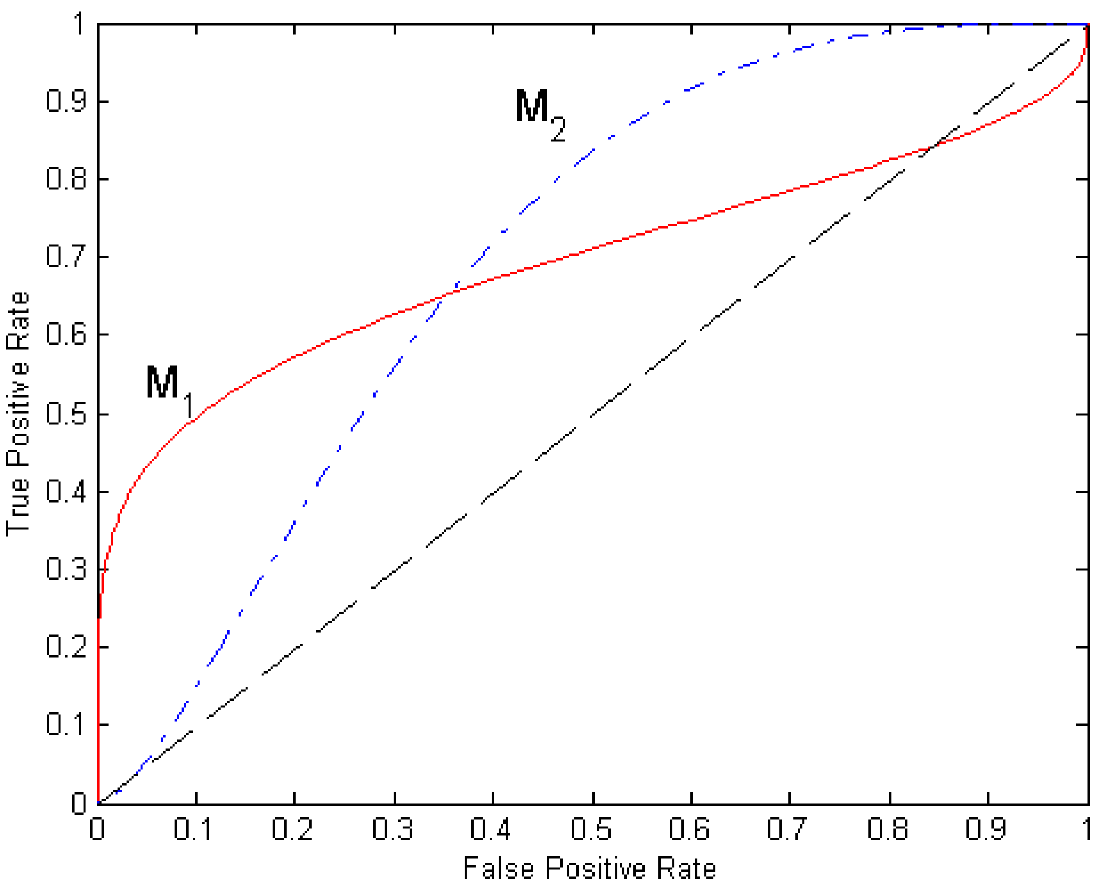
Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
→ TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
→ FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

ROC Curve:



Instance	Score	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

Using ROC for Model Comparison

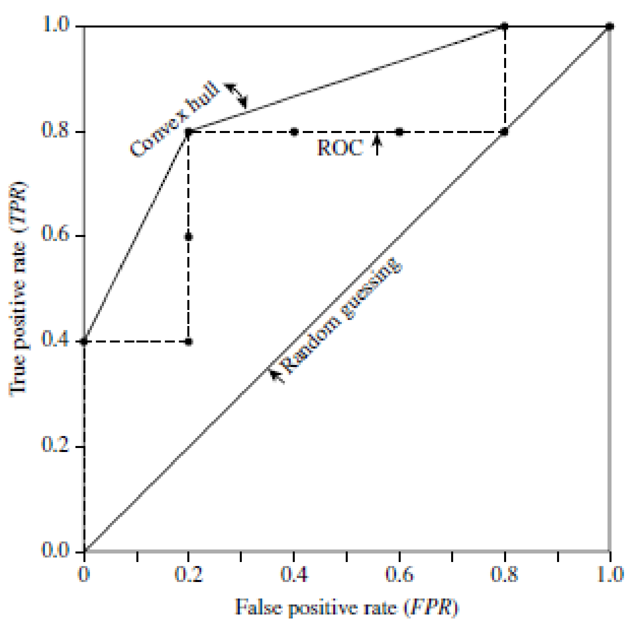


- No model consistently outperforms the other
  - M_1 is better for small FPR
  - M_2 is better for large FPR
- Area Under the ROC curve (AUC)
  - Ideal:
- Area = 1
  - Random guess:
- Area = 0.5
- Higher the AUC, better the model is at predicting

Another Example

The table below shows the probability value (column 3) returned by a probabilistic classifier for each of the 10 tuples in a test set, sorted by decreasing probability order. The corresponding ROC is given on right hand side.

Tuple #	Class	Prob.	TP	FP	TN	FN	TPR	FPR
1	P	0.90	1	0	5	4	0.2	0
2	P	0.80	2	0	5	3	0.4	0
3	N	0.70	2	1	4	3	0.4	0.2
4	P	0.60	3	1	4	2	0.6	0.2
5	P	0.55	4	1	4	1	0.8	0.2
6	N	0.54	4	2	3	1	0.8	0.4
7	N	0.53	4	3	2	1	0.8	0.6
8	N	0.51	4	4	1	1	0.8	0.8
9	P	0.50	5	4	0	1	1.0	0.8
10	N	0.40	5	5	0	0	1.0	1.0



Common Issues in Classifiers

Class Imbalance Problem

Problems where the classes are skewed (more records from one class than another)

- Find needle in haystack
- Lots of classification problems where the classes are skewed (more records from one class than another)
  - Credit card fraud
  - Intrusion detection
  - Defective products in manufacturing assembly line

Simple Techniques to Solve :

- Up-sample minority class
  - randomly duplicating observations from a minority class
- Down-sample majority class
  - removing random observations.
- Generate Synthetic Samples
  - new samples based on the distances between the point and its nearest neighbors

Class Imbalance Problem

- The main class of interest is **rare**.
- The data set distribution reflects a significant majority of the negative class (Eg., Job Offered = Yes/1) and a minority positive class (Eg., Job Offered = No/0)
- For Another Example,
  - fraud detection applications, the class of interest (or positive class) is *"fraud,"*
  - *medical tests, there may be a rare class, such as "cancer"*
- **Accuracy might not be a good option** for measuring performance in case of class imbalance problem

Popular Solutions to Class Imbalance

- Generate Synthetic Samples
- New samples based on the distances between the point and its nearest neighbors E.g. Synthetic Minority Oversampling Technique, or SMOTE class in sklearn
- Change the performance metric : Use Recall, Precision or ROC curves instead of accuracy
- Try different algorithms : Some algorithms as Support Vector Machines and Tree-Based algorithms may work better with imbalanced classes. We will discuss these post mid term

Many measures exists, but none of them may be ideal in all situations.
Significant Factors that help :

- Level of class imbalance
- Importance of TP vs FP
- Cost/Time tradeoffs

Dealing with Imbalanced Classes - Summary

- Many measures exists, but none of them may be ideal in all situations
  - Random classifiers can have high value for many of these measures
  - TPR/FPR provides important information but may not be sufficient by itself in many practical scenarios
  - Given two classifiers, sometimes you can tell that one of them is strictly better than the other
    - C1 is strictly better than C2 if C1 has strictly better TPR and FPR relative to C2 (or same TPR and better FPR, and vice versa)
  - Even if C1 is strictly better than C2, C1's F-value can be worse than C2's if they are evaluated on data sets with different imbalances
  - Classifier C1 can be better or worse than C2 depending on the scenario at hand

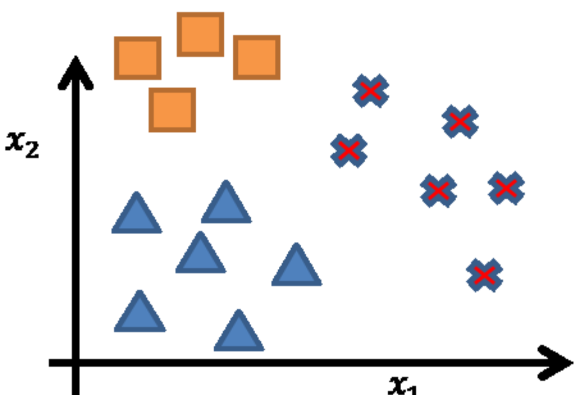
Types of Classification Based on the Output Labels

Types of Classification

Output Labels

- Target Concept

Multi Class

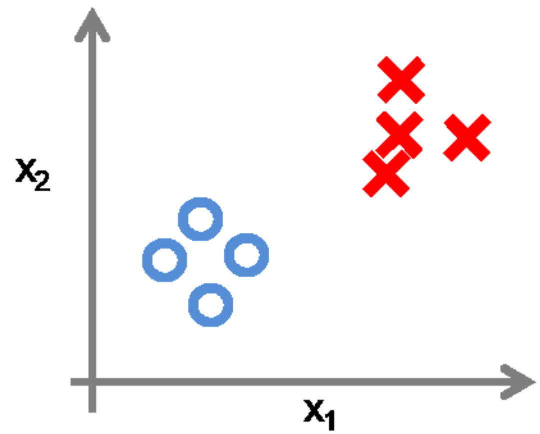


Enjoy Sports

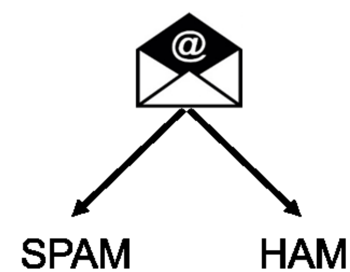
Examples of Multiclass:

- Email foldering /tagging: Work, Friends, Family, Hobby
- Medical Diagnostics: Not ill, Cold, Flu
- Weather: Sunny, Cloudy, Rain, Snow

Binary

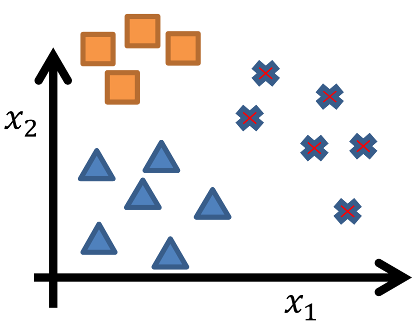


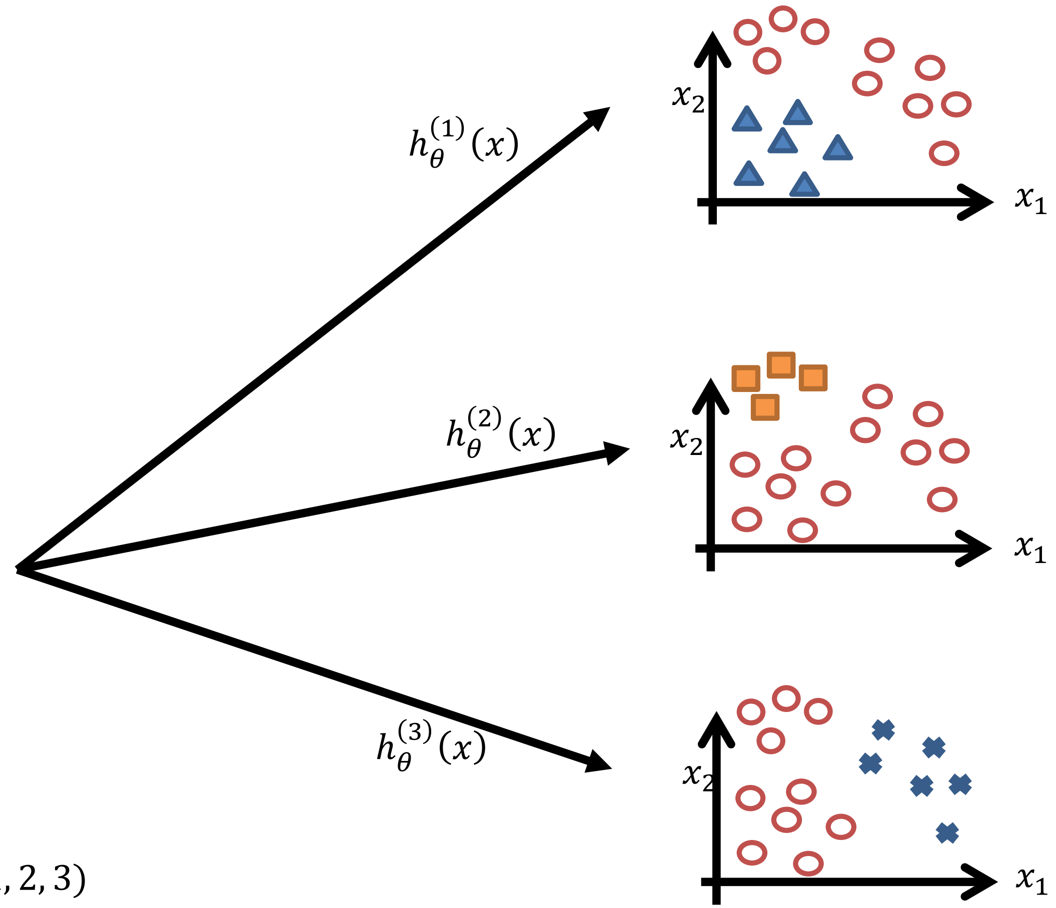
Spam Classifier



Prediction – Multi class Classification

One Vs All Strategy(one-vs-rest)





Class 1:
Class 2:
Class 3:

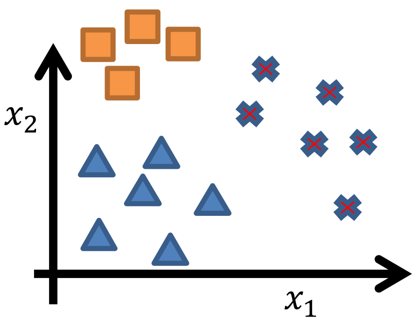
$$h_\theta^{(i)}(x) = P(y = i|x;\theta) \quad (i = 1, 2, 3)$$

For input x Predict : $\max\limits_{i} h_\theta^{(i)}(x)$

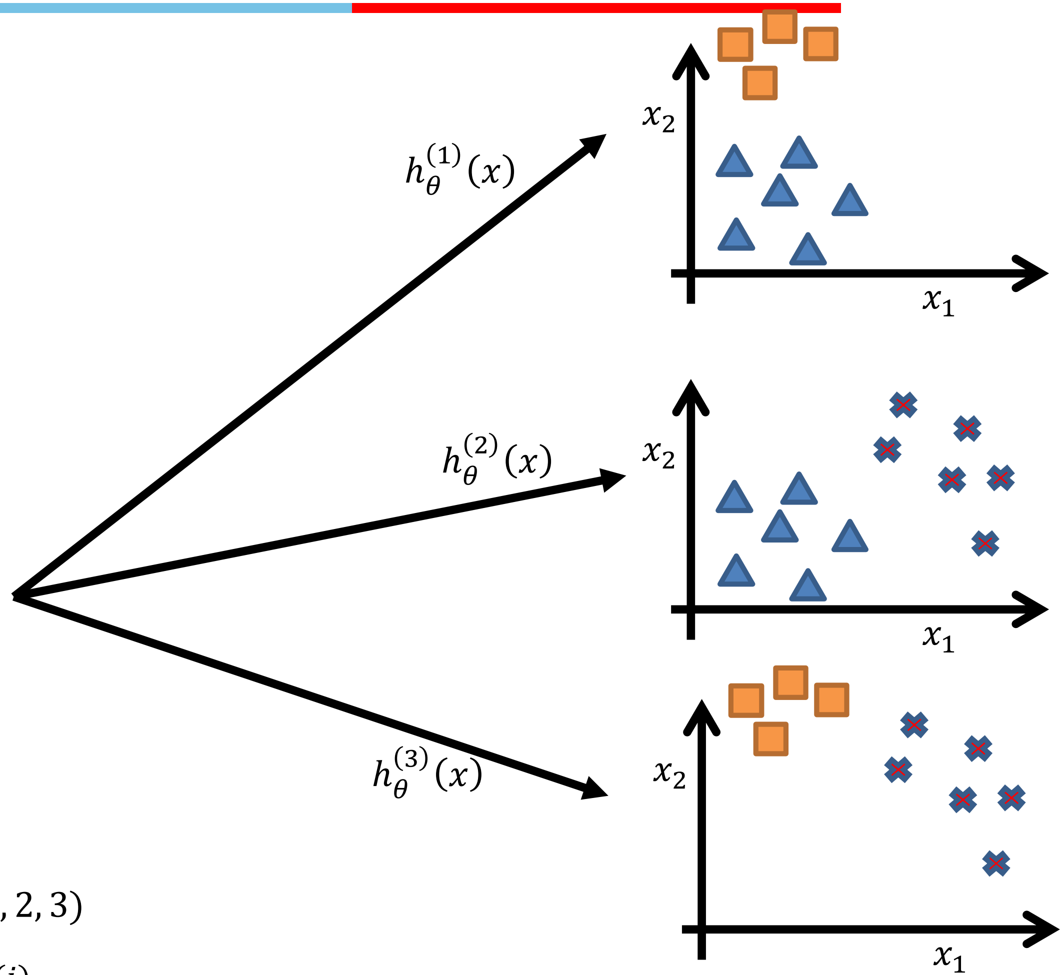
Note: Scikit-Learn detects when you try to use a binary classification algorithm for a multi-class classification task, and it automatically runs OvA (except for SVM classifiers for which it uses OvO)

Prediction – Multi class Classification

One Vs One Strategy



Class 1: ▲
Class 2: ■
Class 3: ✖



$$h_\theta^{(i)}(x) = P(y = i|x;\theta) \quad (i = 1, 2, 3)$$

For input x Predict : $\max_i h_\theta^{(i)}(x)$

N × (N – 1) / 2 classifiers

Logistic regression (Classification)-Summary

- **Model**

$$h_\theta(x) = P(Y = 1|X_1, X_2, \cdots, X_n) = \frac{1}{1+e^{-\theta^\top x}}$$

- **Cost function**

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})) \qquad \mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

- **Learning**

Gradient descent: Repeat $\{\theta_j := \theta_j - \alpha \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}\}$

- **Inference**

$$\hat{Y} = h_\theta(x^{\text{test}}) = \frac{1}{1 + e^{-\theta^\top x^{\text{test}}}}$$

Note:

- σ(t) < 0.5 when t < 0, and σ(t) ≥ 0.5 when t ≥ 0, so a Logistic model predicts 1 if xTθ is positive, and 0 if it is negative
- logit(p) = log(p / (1 - p)), is the inverse of the logistic function. Indeed, if you compute the logit of the estimated probability p, you will find that the result is t. The logit is also called the log-odds

Logistic Regression –Additional Practice Exercises

CGPA	IQ	IQ	Job Offered
5.5	6.7	100	1
5	7	105	0
8	6	90	1
9	7	105	1
6	8	120	0
7.5	7.3	110	0

Hyper parameters:
Learning Rate = 0.8
Initial Weights = (-0.1, 0.2,-0.5)
Regularization Constant = 10

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^\top x}}$$

For this similar problem discussed in class note that the hyper parameters are different

1. Formulate the gradient descent update equations for this problem
2. Repeat the GD for two iterations
3. Find the Loss at every iterations and interpret your observation
4. Using the results of second iteration answer below questions:
    a) Interpret the influence of the CGPA in the response variable
    b) Predict if a new candidates with IQ=5 and CGPA = 9 will be offered job or not?
5. Repeat the steps 2 to 4 by using stochastic gradient descent instead of batch gradient descent for 4 iterations. (Take any random sample from among 6 instances for these 4 iterations)

Evaluation of Classifiers–Additional Practice Exercises

Given below is a confusion matrix for medical data where the class values are yes and no for a class label attribute, cancer. Answer the following questions.

Classes	*yes*	*no*	*Total*	*Recognition (%)*
yes	**90**	**210**	300	30.00
no	**140**	**9560**	9700	98.56
Total	230	9770	10,000	96.40

Confusion matrix for the classes *cancer* = *yes* and *cancer* = *no*.

1. Calculate the Precision , Recall, F-Score, Error-rate, F-Score
2. Brainstorm on the use case / scenarios w.r.t given example, where precision is preferred over recall.
3. Brainstorm on the use case / scenarios w.r.t given example, where recall is preferred over precision.

Formulation of the Gradient Descent equation for Logistic regression from its cross entropy loss function (Additional Reference for student's Self Reading)

Logistic regression GD derivation

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^T \boldsymbol{x}}}$$

$$\sigma(\mathrm{x}) = \frac{1}{1 + e^{-\mathrm{x}}}$$

$$\frac{d\big(\sigma(\mathrm{x})\big)}{dx} = \frac{0 * (1 + e^{-\mathrm{x}}) - (1) * \big(e^{-\mathrm{x}} * (-1)\big)}{(1 + e^{-\mathrm{x}})^2}$$

$$\frac{d\big(\sigma(\mathrm{x})\big)}{dx} = \frac{(e^{-\mathrm{x}})}{(1 + e^{-\mathrm{x}})^2} = \frac{1\text{-}1 + (e^{-\mathrm{x}})}{(1 + e^{-\mathrm{x}})^2} = \frac{1 + e^{-\mathrm{x}}}{(1 + e^{-\mathrm{x}})^2} - \frac{1}{(1 + e^{-\mathrm{x}})^2}$$

$$\frac{d\big(\sigma(\mathrm{x})\big)}{dx} = \frac{1}{1 + e^{-\mathrm{x}}} * \left(1 - \frac{1}{1 + e^{-\mathrm{x}}}\right) = \sigma(\mathrm{x})\big(1 - \sigma(\mathrm{x})\big)$$

Logistic regression cost function

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

$$\Downarrow$$

- $\text{Cost}(h_\theta(x), y) = -y\ \log\big(\mathrm{h}_\theta(\mathrm{x})\big) - (1 - \mathrm{y})\log\big(1 - h_\theta(x)\big)$

- If $y = 1$: $\text{Cost}(h_\theta(x), y) = -\log\big(h_\theta(x)\big)$
- If $y = 0$: $\text{Cost}(h_\theta(x), y) = -\log\big(1 - h_\theta(x)\big)$

Slide credit: Andrew Ng

Step –I

Applying Chain rule and writing in terms of partial derivatives

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * \sum_{i=1}^{m}\left[y^{(i)} * \frac{1}{h_\theta(x^{(i)})} * \frac{\partial\left(h_\theta(x^{(i)})\right)}{\partial(\theta j)}\right]$$

$$+ \sum_{i=1}^{m}\left[\left(1 - y^{(i)}\right) * \frac{1}{\left(1 - h_\theta(x^{(i)})\right)} * \frac{\partial\left(1 - h_\theta(x^{(i)})\right)}{\partial(\theta j)}\right]$$

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * \left(\sum_{i=1}^{m}\left[y^{(i)} * \frac{1}{h_\theta(x^{(i)})} * \sigma(z)\left(1 - \sigma(z)\right) * \frac{\partial(\theta^T x)}{\partial(\theta j)}\right]\right.$$

$$\left. + \sum_{i=1}^{m}\left[\left(1 - y^{(i)}\right) * \frac{1}{\left(1 - h_\theta(x^{(i)})\right)} * (-\sigma(z)\left(1 - \sigma(z)\right) * \frac{\partial(\theta^T x)}{\partial(\theta j)}\right]\right)$$

Step –II

- Evaluating the partial derivative using the pattern of the derivative of the sigmoid function.

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * (\sum_{i=1}^{m} \left[y^{(i)} * \frac{1}{h_\theta(x^{(i)})} * \sigma(z)(1-\sigma(z)) * \frac{\partial(\theta^T x)}{\partial(\theta j)} \right]$$

$$+ \sum_{i=1}^{m} \left[(1-y^{(i)}) * \frac{1}{(1-h_\theta(x^{(i)}))} * (-\sigma(z)(1-\sigma(z))^* \frac{\partial(\theta^T x)}{\partial(\theta j)} \right])$$

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * (\sum_{i=1}^{m} \left[y^{(i)} \frac{1}{h_\theta(x^{(i)})} h_\theta(x^{(i)}) \left(1-h_\theta(x^{(i)})\right) * x_j^i \right] +$$

$$\sum_{i=1}^{m} \left[(1-y^{(i)}) * \frac{1}{(1-h_\theta(x^{(i)}))} * (-h_\theta(x^{(i)}) \left(1-h_\theta(x^{(i)})\right) * x_j^i \right])$$

Step –III

- Simplifying the terms by multiplication

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * (\sum_{i=1}^{m} \left[y^{(i)} * \left(1 - h_\theta(x^{(i)})\right) * x_j^i - (1 - y^{(i)}) * h_\theta(x^{(i)}) * * x_j^i \right]$$

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * (\sum_{i=1}^{m} [\, y^{(i)} - y^{(i)} * h_\theta(x^{(i)}) - h_\theta(x^{(i)}) + y^{(i)} * h_\theta(x^{(i)})] * x_j^i\,)$$

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * (\sum_{i=1}^{m} [\, y^{(i)} - h_\theta(x^{(i)})] * x_j^i\,)$$

Additional References

- Tom M. Mitchell
  Generative and discriminative classifiers: Naïve Bayes and Logistic Regression
  http://www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf

- Andrew Ng, Michael Jordan
  On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes
  http://papers.nips.cc/paper/2020-on-discriminative-vs-generative-classifiers-a-comparison-of-logistic-regression-and-naive-bayes.pdf
- http://www.cs.cmu.edu/~tom/NewChapters.html
- http://ai.stanford.edu/~ang/papers/nips01-discriminativegenerative.pdf
- https://medium.com/@sangha_deb/naive-bayes-vs-logistic-regression-a319b07a5d4c
- https://www.youtube.com/watch?v=-la3q9d7AKQ
- http://www.datasciencesmachinelearning.com/2018/11/handling-outliers-in-python.html

Interpretability

- https://christophm.github.io/interpretable-ml-book/logistic.html

Thank you !

Required Reading for completed session :

T1 - Chapter # 6 (Tom M. Mitchell, Machine Learning)

R1 – Chapter # 3,#4 (Christopher M. Bhisop, Pattern Recognition & Machine Learning) & Refresh your MFDS course basics

Next Session Plan :

Module 5 – Decision Tree Classifier