Welcome!!

Machine Learning

AIML CZG565

Linear Models for Classification

Dr. Sugata Ghosal

sugata.ghosal@pilani.bits-pilani.ac.in

BITS Pilani

Pilani Campus

Agenda

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

Decision Theory
&
Classification Models

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]

f(x)

discriminative classifiers

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]

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 :

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

Sky	AirTemp	Humidity	Wind	Water	Forecast	*P(EnjoySport =Yes)*
X =<Sunny ,	Warm ,	Normal ,	Strong ,	Warm ,	Same > →	0.95 = $P(C_k \mid X)$
Sunny	Warm	High	Strong	Warm	Same	0.7
Rainy	Cold	High	Strong	Warm	Change	0.5
Sunny	Warm	High	Strong	Cool	Change	0.6

Classification Problem: Stages

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

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

Sky	AirTemp	Humidity	Wind	Forecast	Enjoy Sport?
Sunny	Warm	Normal	Strong	Same	Yes
Sunny	Warm	High	Strong	Same	Yes
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	High	Strong	Change	Yes

Training Set

Learning algorithm

Induction/ Inference step

Learn Model for $p(x,C_k)$

Model

Apply Model to find optimal t

Deduction/ Decision Step

Sky	AirTemp	Humidity	Wind	Forecast	Enjoy Sport?
Rainy	Cold	High	Strong	Change	?
Sunny	Warm	High	Strong	Change	?
Rainy	Warm	Normal	Breeze	Same	?

Test Set

Decision Region

Sky	AirTemp	Humidity	Wind	Forecast	Enjoy Sport?
Sunny	Warm	Normal	Strong	Same	Yes
Sunny	Warm	High	Strong	Same	Yes
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	High	Strong	Change	Yes

Training Set

Learning algorithm

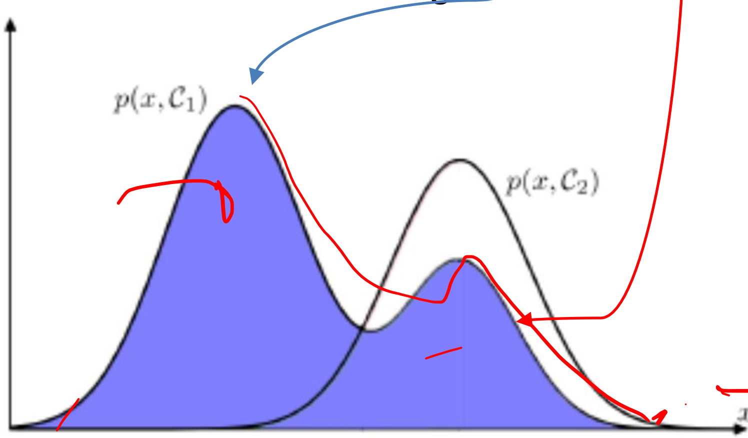
Induction/ Inference step

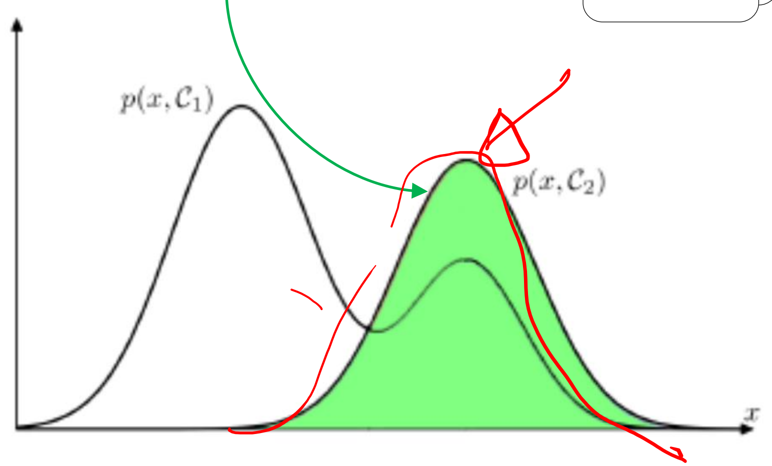
Learn Model for $p(x,C_k)$

Model

Model divides the input space into regions $\mathbf{R}_k$ 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

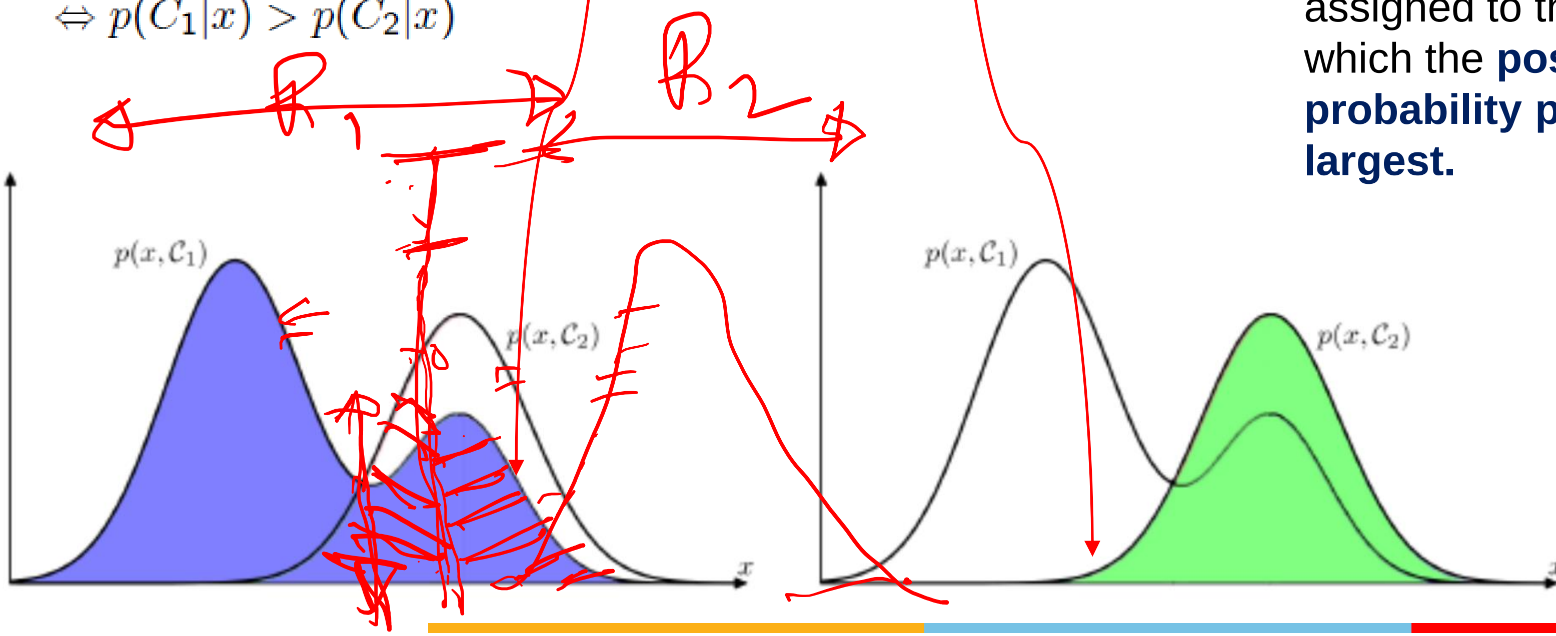
$$p(C_k|x) = \frac{p(x, C_k)}{p(x)}$$

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

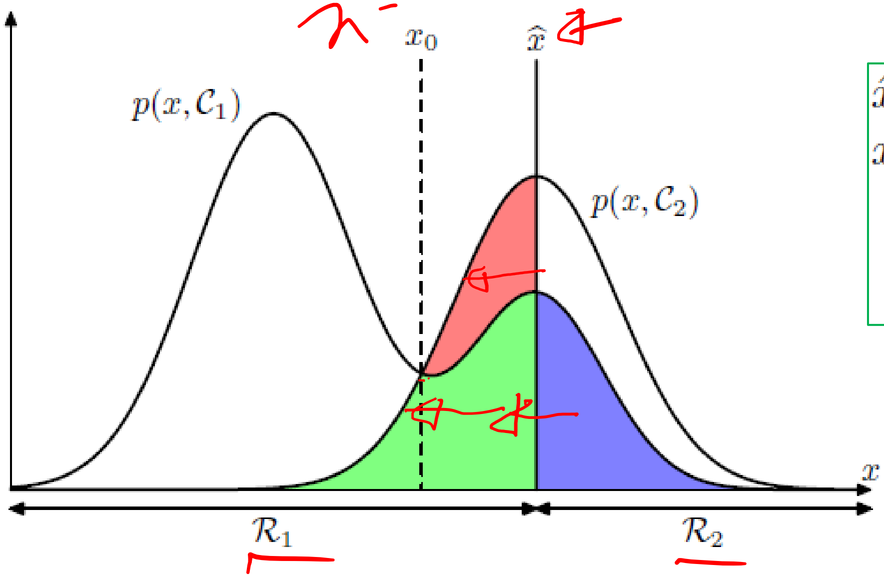
To minimize p(mistake), each **x** is assigned to whichever class has the smaller value of the integrand

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.**

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



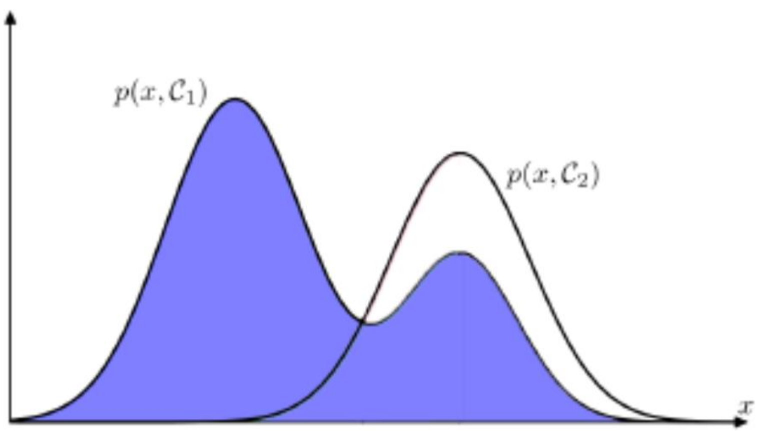
Decision Theory - Summary

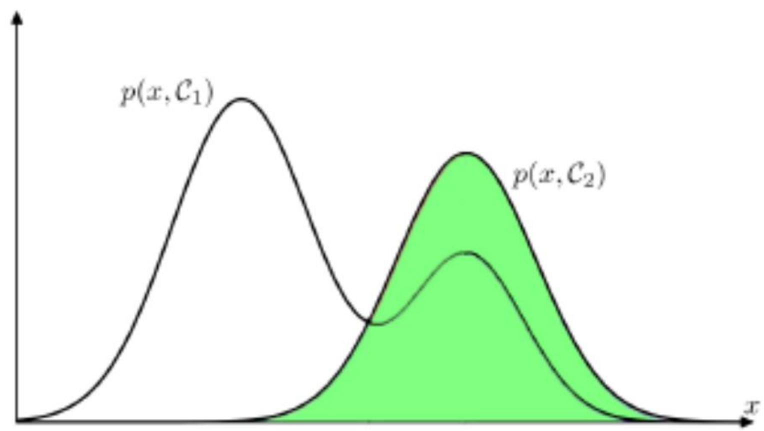


$\hat{x}$: decision boundary.

x_0: optimal decision boundary

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





Linear Models for Classification

Types of Classification

Inductive Learning Hypothesis : Interpretation

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

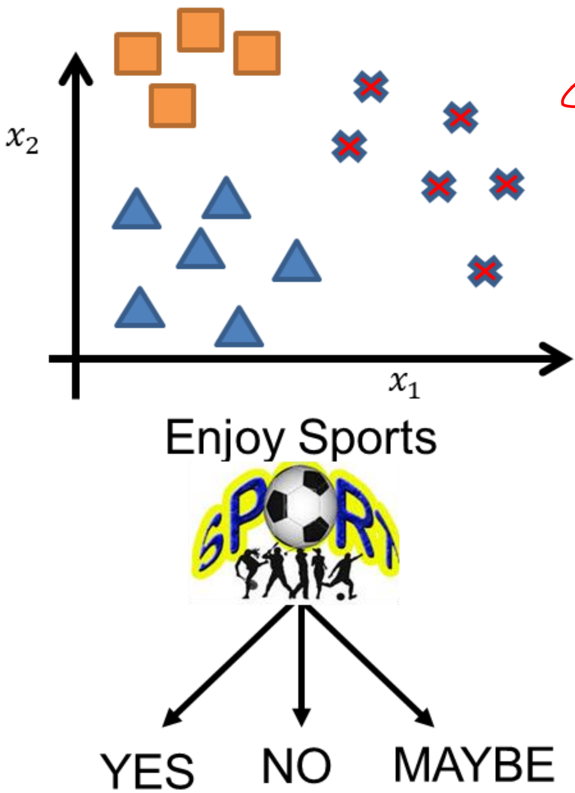
Sky	AirTemp	Altitude	Wind	Water	Forecast	Humidity
Sunny	Warm	Normal	Strong	Warm	Same	60
Sunny	Warm	High	Strong	Warm	Same	75
Rainy	Cold	High	Strong	Warm	Change	70
Sunny	Warm	High	Strong	Cool	Change	45

Types of Classification

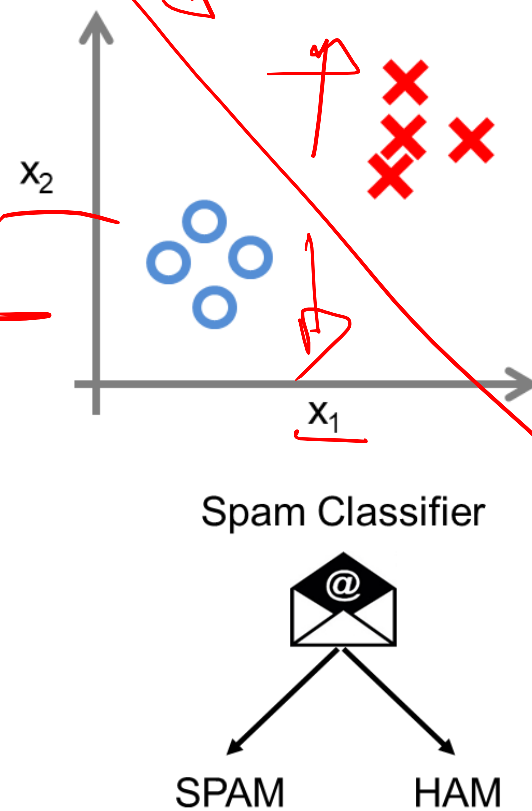
Output Labels

- Target Concept

Multi Class



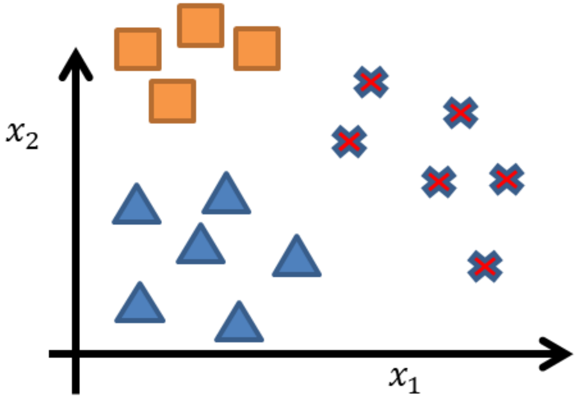
Binary



Output Labels

- Target Concept

Multi Class



Prediction – Multi class Classification

One Vs All Strategy(one-vs-rest)

1-2

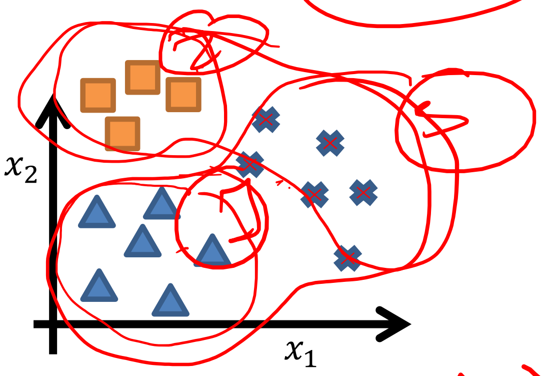
(1) or 2

2-3

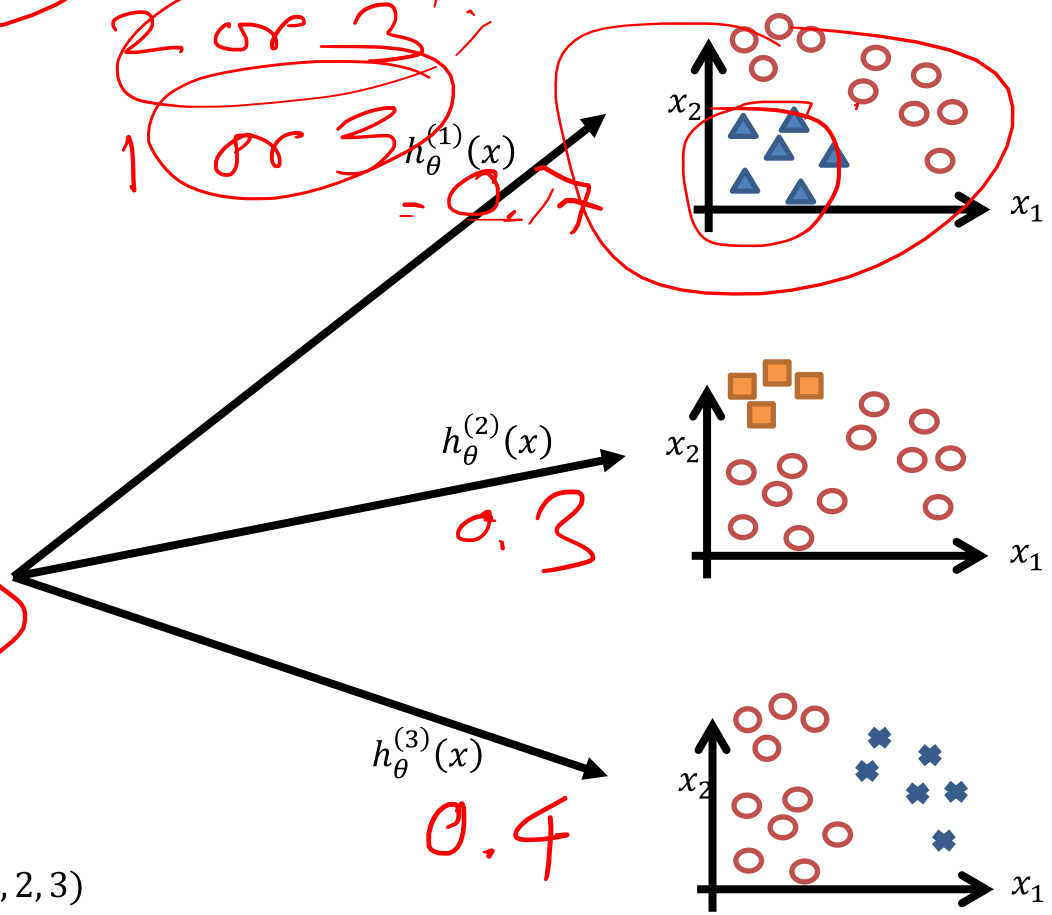
2 or 3

1-3

1 or 3



Class 1: ▲
Class 2: ■
Class 3: ✖



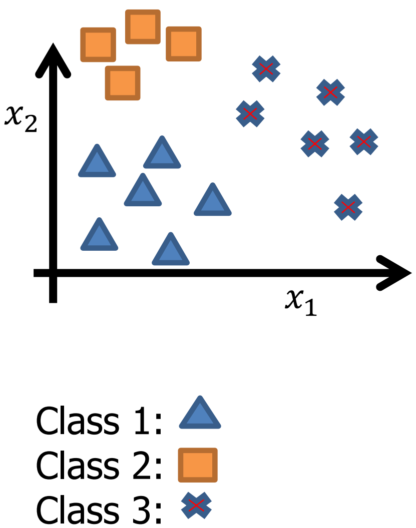
$$h_\theta^{(i)}(x) = P(y = i|x;\theta) \quad (i = 1, 2, 3)$$

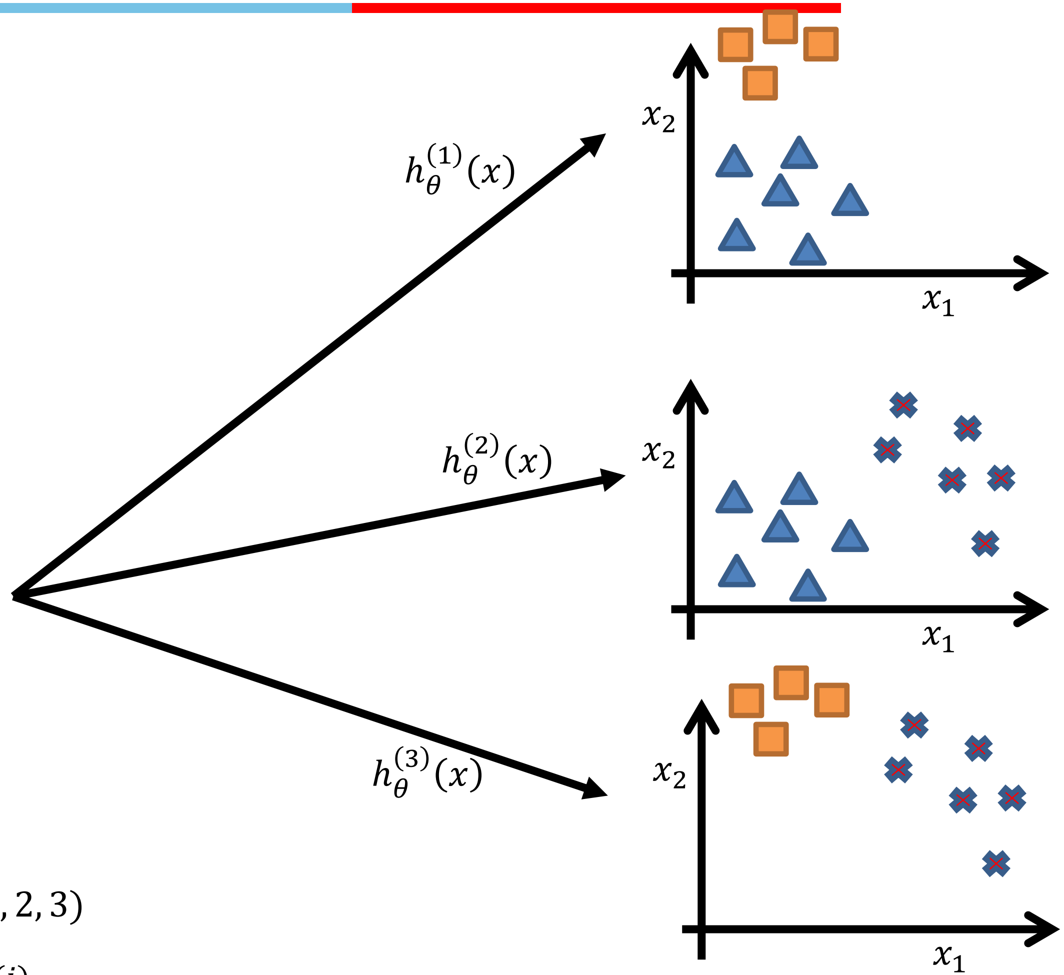
$$\text{For input x Predict}: \max_i h_\theta^{(i)}(x)$$

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





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

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

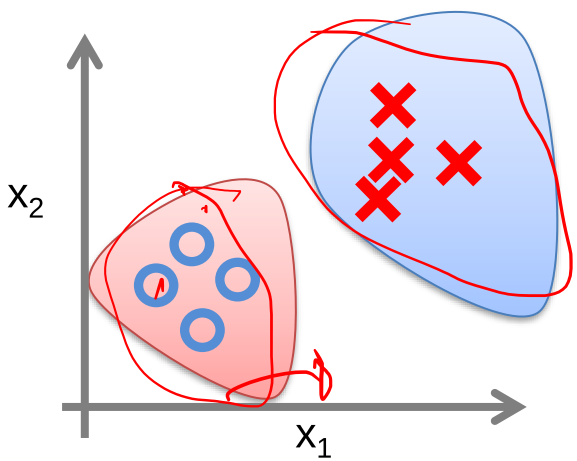
N × (N – 1) / 2 classifiers

Types of Classification

Decision Theory: Interpretation

Model Building

Generative



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

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

Eg., Gaussians, **Naïve Bayes**, Mixtures of multinomials , **Mixtures of Gaussians**, Bayesian networks

Likelihood

Class Prior Probability

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

Posterior Probability

Predictor Prior Probability

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

Sky	AirTemp	Humidity	Wind	Forecast	Enjoy Sport?
Sunny	Warm	Normal	Strong	Same	Yes
Sunny	Warm	High	Strong	Same	No
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	Normal	Breeze	Same	Yes
Sunny	Hot	Normal	Breeze	Same	No
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	High	Strong	Change	Yes
Rainy	Warm	Normal	Breeze	Same	Yes

f(x) > 0.5 for all points x
< 0.5 for 0

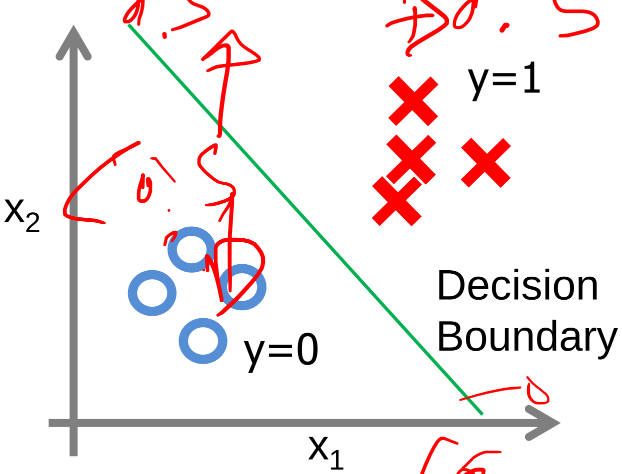
Decision Theory: Interpretation

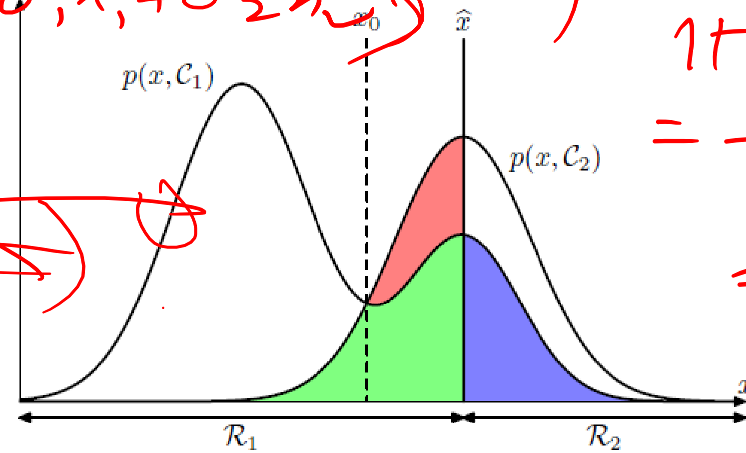
Model Building

$f(x) = \frac{1}{1+e^{-(\theta_0+\theta_1 x_1+\theta_2 x_2 \ldots)}} = \frac{1}{1+e^{0}} = \frac{1}{1+1} = 0.5$

> 0.5

Discriminative





$(\theta_0+\theta_1 x_1 + \theta_2 x_2 > 0)$

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

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

$P(c \mid x)$

Posterior Probability

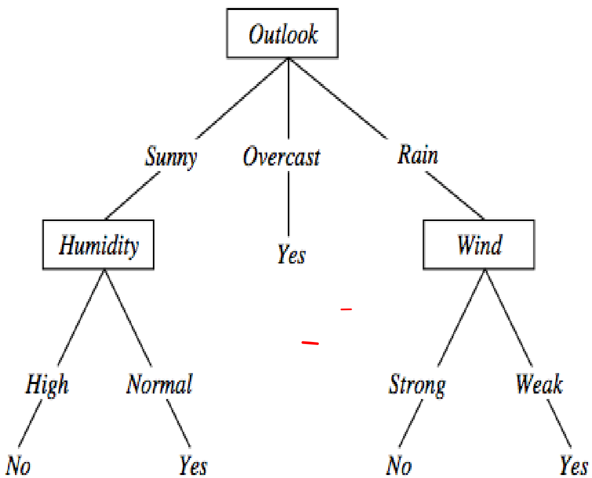
Sky	AirTemp	Humidity	Wind	Forecast	Enjoy Sport?
Sunny	Warm	Normal	Strong	Same	Yes
Sunny	Warm	High	Strong	Same	No
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	Normal	Breeze	Same	Yes
Sunny	Hot	Normal	Breeze	Same	No
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	High	Strong	Change	Yes
Rainy	Warm	Normal	Breeze	Same	Yes

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

Types of Classification

Decision Theory: Interpretation

Model Building



$$IF\ OUTLOOK = Overcast\ THEN\ PLAY = Yes$$

ELSE

$$IF\ OUTLOOK = Rain\ AND\ \ WIND = Strong\ THEN\ PLAY = No$$

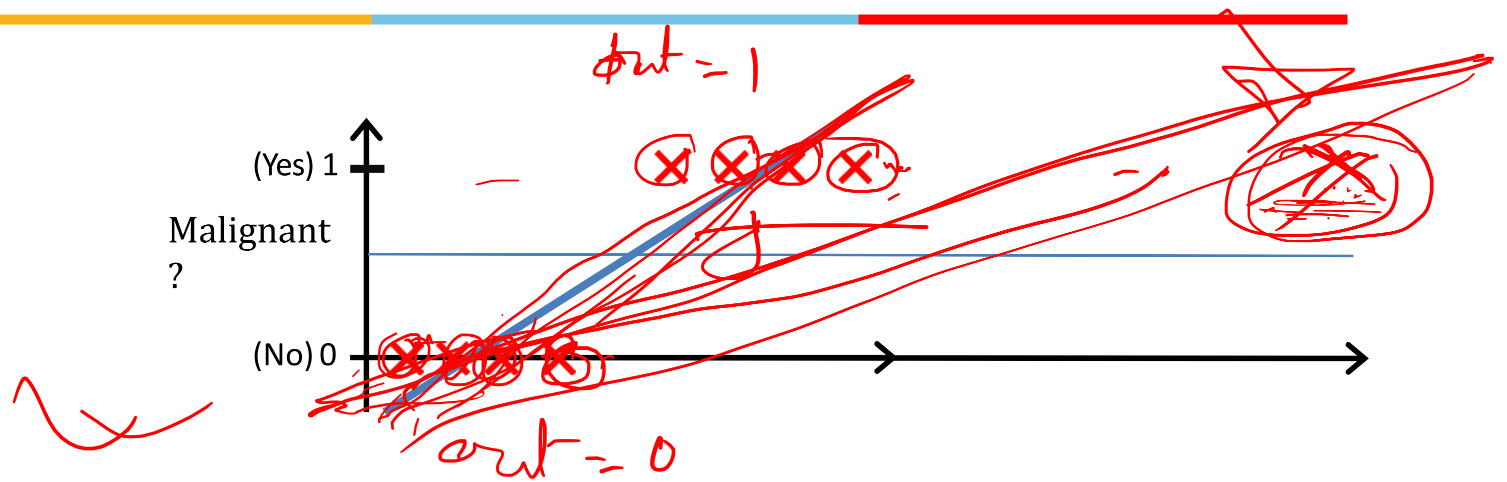
Sky	AirTemp	Humidity	Wind	Forecast	Enjoy Sport?
Sunny	Warm	Normal	Strong	Same	Yes
Sunny	Warm	High	Strong	Same	No
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	Normal	Breeze	Same	Yes
Sunny	Hot	Normal	Breeze	Same	No
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	High	Strong	Change	Yes
Rainy	Warm	Normal	Breeze	Same	Yes

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

Logistic Regression

BITS Pilani

Logistic Regression vs Least Squares Regression



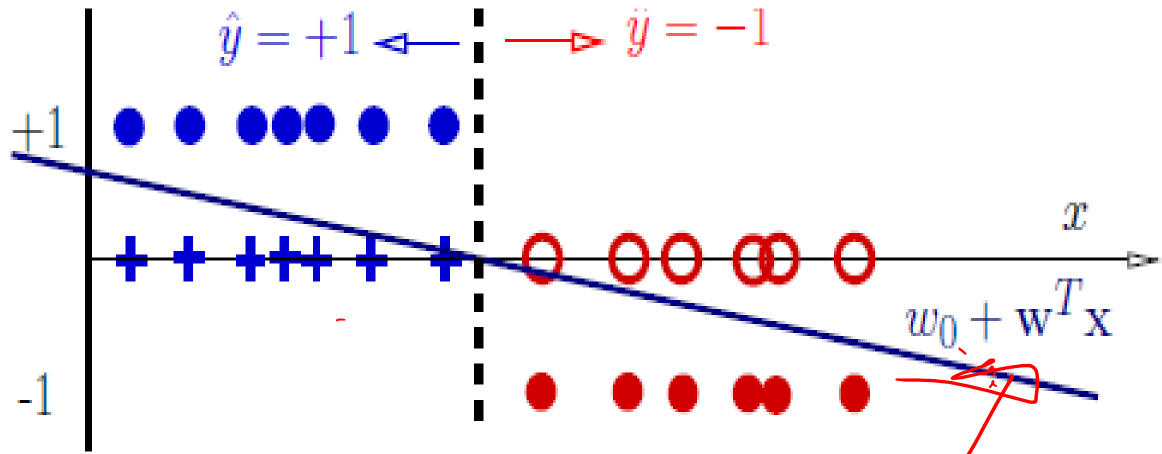
- Tumor Size
- Can we solve the problem using linear regression? E.g., fit a straight line and define a threshold at 0.5
- Threshold classifier output $h_\theta(x)$ at 0.5:

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

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

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

Decision Rules



- Classifier:

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

- Decision rule is

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

$f(x) = \frac{1}{1 + e^{-(w_0 + w_1 \ldots + w_d)}}$

0.5

- 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.

Learning model parameters

- Training set:

$$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$$

- m examples

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

$\theta^T x = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots$

$h_\theta(x^{(m)}) \approx y^{(m)}$

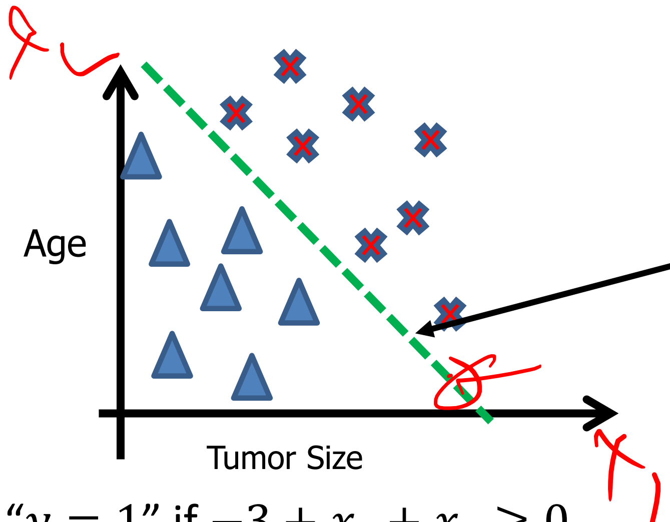
$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

$\approx y^{(m)}$

- How to choose parameters (feature weights) ?

Logistic Regression

- 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., $\theta_0 = -3,\ \theta_1 = 1,\ \theta_2 = 1$



$-3 + \text{TumorSize} + \text{Age} = 0$

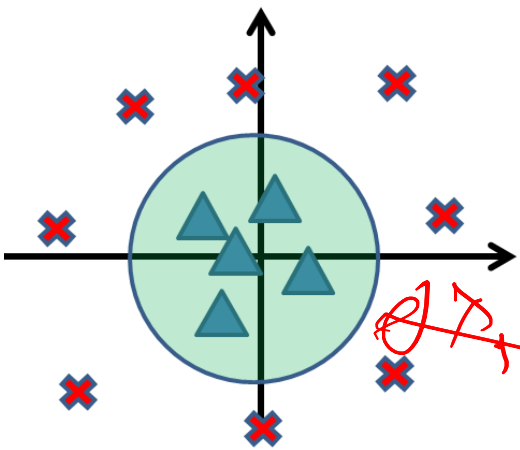
- Predict "$y = 1$" if $-3 + x_1 + x_2 \geq 0$

Slide credit: Andrew Ng

Logistic Regression

$(x_1, x_2) \to (x_1, x_2, x_1^2, x_2^2)$

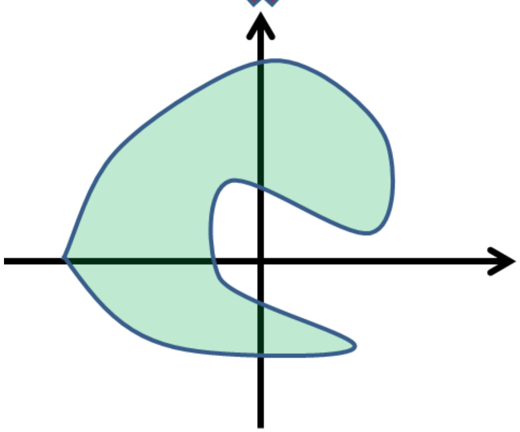
LR



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

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

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



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

Slide credit: Andrew Ng

Logistic Regression

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

- m examples
- n features

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

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

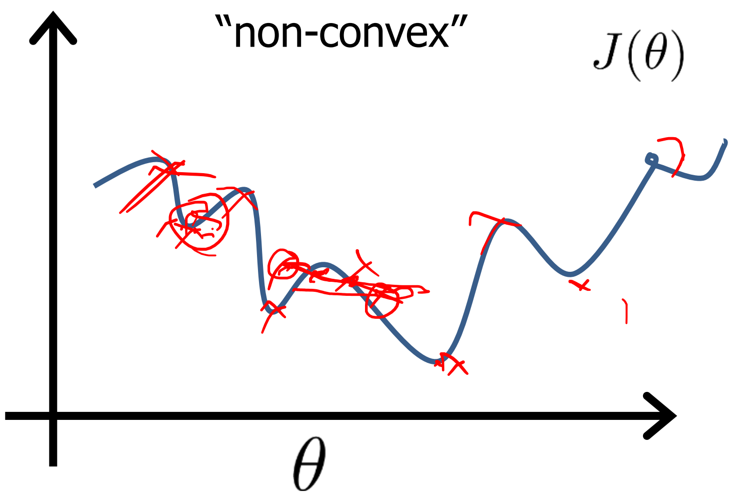
- How to choose parameters (feature weights)? θ

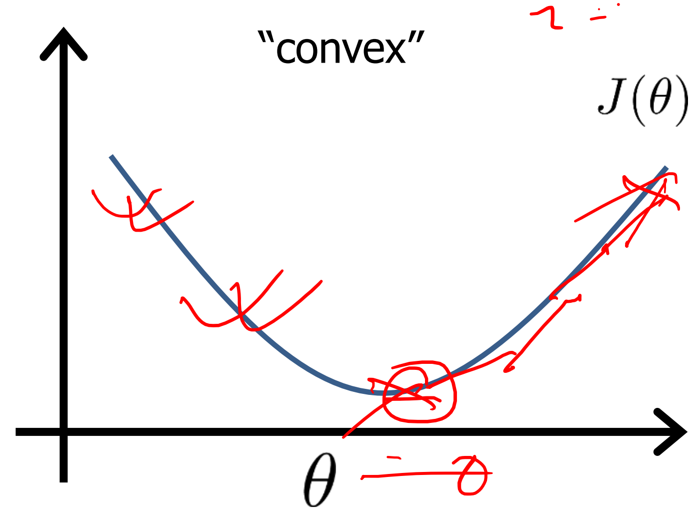
Logistic Regression

$\hat{y}_i = \dfrac{1}{1+e^{-\theta_0 - \theta_1 x_1}}$

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

$\sum_{i=1} (\hat{y}_i - y_i)^2$





θ_0, θ_1

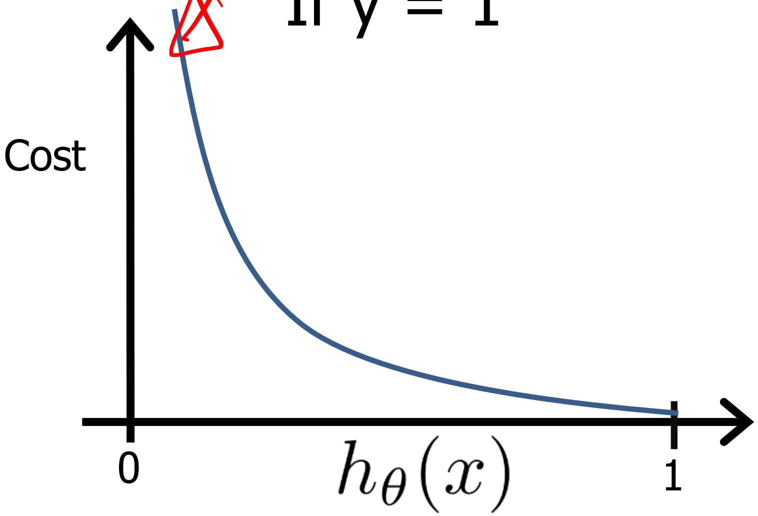
Log-loss

Logistic regression cost function (cross entropy)

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

$-\log(h_\theta(x))$

If y = 1

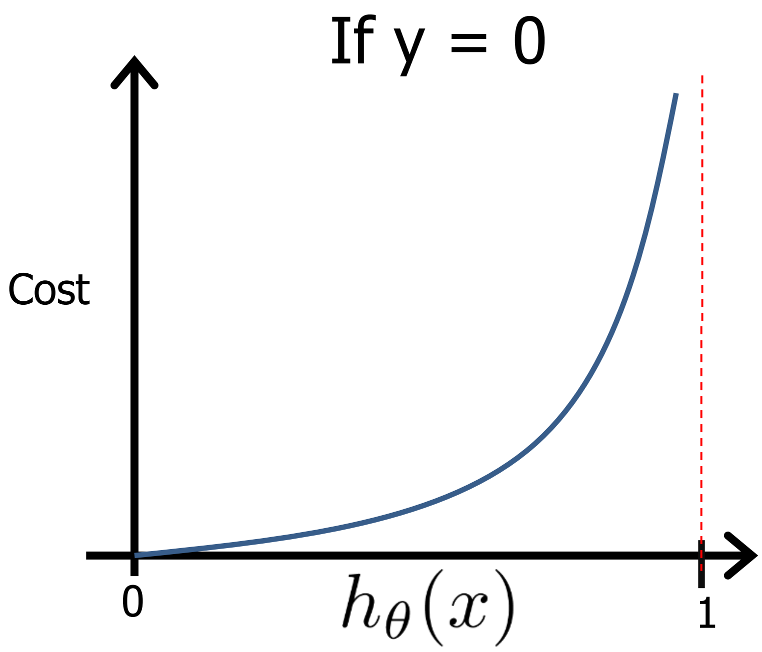


$\mathrm{Cost} = 0$ if $y = 1, h_\theta(x) = 1$
But as $h_\theta(x) \to 0$
$Cost \to \infty$

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

average loss fn. value over training set

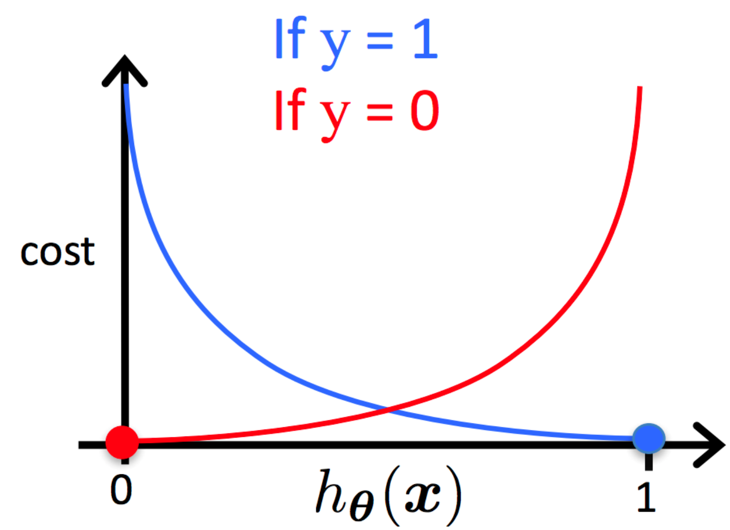
$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

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

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

To make a prediction given new x :

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



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

{

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

}

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

target

Gradient Descent Algorithm

$h_\theta(x)$

$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots$

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 {

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

}

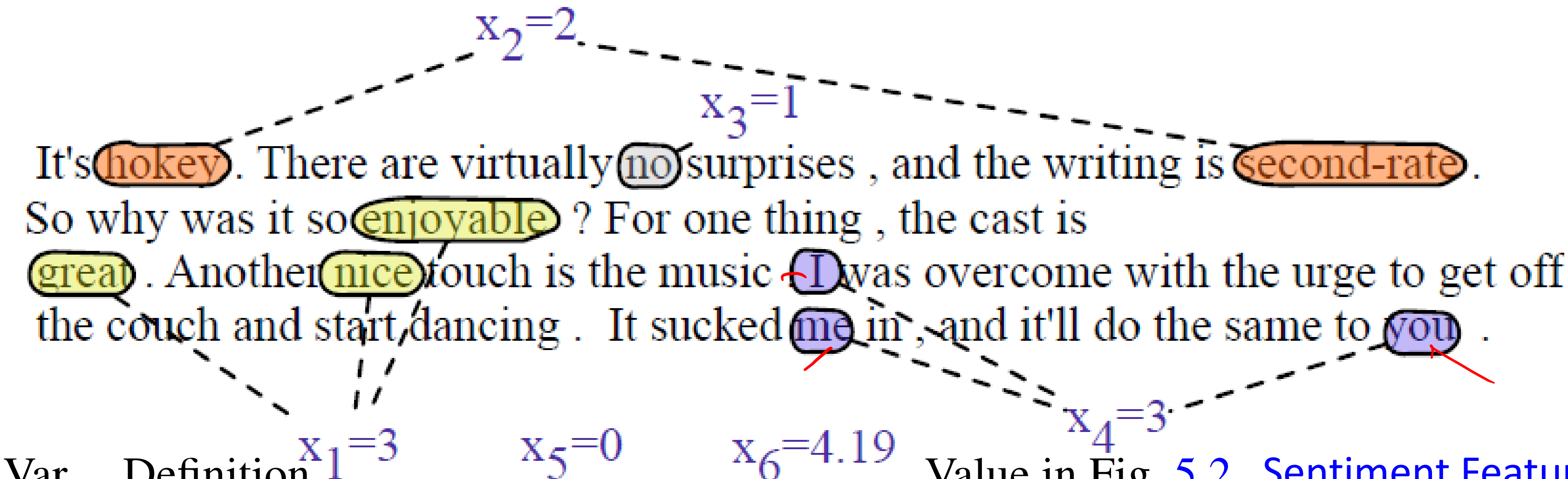
$$\boxed{h_\theta(x) = \frac{1}{1 + e^{-\theta^\top x}}}$$

$$h_\theta(x) = \frac{1}{1 + e^{-\theta_0 - \theta_1 x_1 - \theta_2 x_2}}$$

Slide credit: Andrew Ng

Example: Sentiment Analysis

NLP



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)	$\ln(66) = 4.19$

Sentiment Features

$[x_1, x_2 \cdots x_6]^T$

Classifying sentiment using logistic regression

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

$\mathrm{b} = 0.1$ $\rightarrow W_0 = \theta_0$

$$\begin{aligned}
p(+|x) = P(Y=1|x) &= \sigma(w \cdot x + b) \rightarrow \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \theta_6 x_6 \\
&= \sigma([2.5, -5.0, -1.2, 0.5, 2.0, 0.7] \cdot [3, 2, 1, 3, 0, 4.19] + 0.1) \\
&= \sigma(.833) \\
&= 0.70
\end{aligned}$$

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

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

$$\sigma(x) = \sigma(0.833) = \frac{1}{1+e^{-0.833}}$$

$0.7 \approx$

Logistic Regression – Fit a Model

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 = 0.3

Initial Weights = (0.5, 0.5,0.5)

Regularization Constant = 0

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

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

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

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

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

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

Approx. : New weights

$\theta_0 = 0.4$

$\theta_{1=CGPA}$ =-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)]

Logistic regression (Classification)

- **Model**

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

- **Cost function**

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

- **Learning**

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

- **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

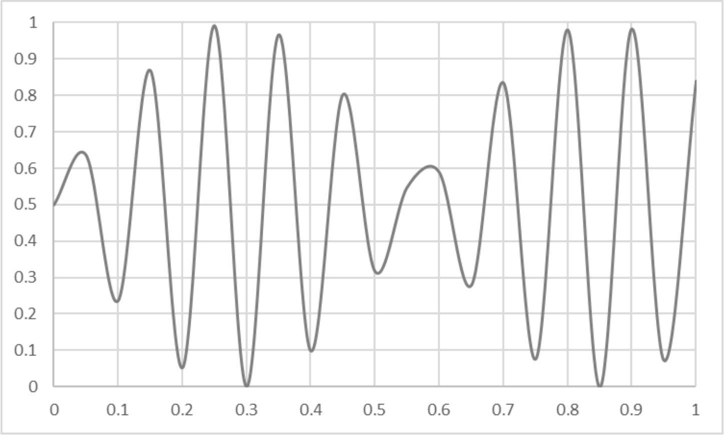
Overfitting vs Underfitting

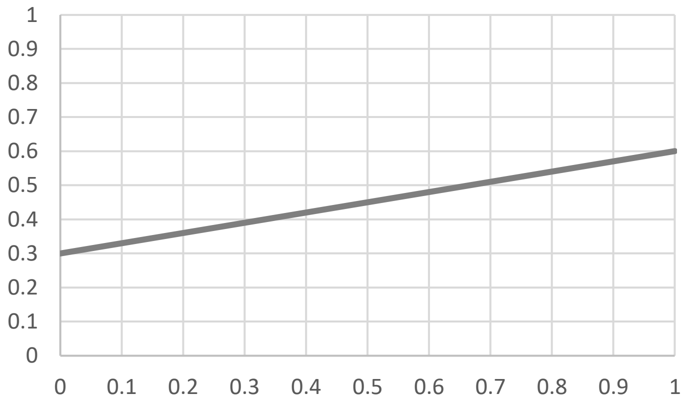
Overfitting

- Fitting the data too well
  - Features are noisy / uncorrelated to concept

Underfitting

- Learning too little of the true concept
  - Features don't capture concept
  - Too much bias in model





Regularization

Note : This topic is already covered in the module 3 . Refresher & few more points added here

Regularization

- A method for automatically controlling the complexity of the learned hypothesis
- **Idea**: penalize for large values of θ_j
  - Can incorporate into the cost function
  - Works well when we have a lot of features, each that contributes a bit to predicting the label
- Can also address overfitting by eliminating features (either manually or via model selection)

Ways to Control Overfitting

- Regularization

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

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.

Regularization

- Linear regression objective function

$$J(\boldsymbol{\theta}) = \underbrace{\frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2}_{\text{model fit to data}} + \underbrace{\frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2}_{\text{regularization}}$$

  - λ is the regularization parameter ($\lambda \geq 0$)
  - No regularization on θ_0!

Understanding Regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2$$

- Note that $\sum_{j=1}^{d}\theta_j^2 = \|\boldsymbol{\theta}_{1:d}\|_2^2$
  - This is the magnitude of the feature coefficient vector!

- We can also think of this as:

$$\sum_{j=1}^{d}(\theta_j - 0)^2 = \|\boldsymbol{\theta}_{1:d} - \vec{\mathbf{0}}\|_2^2$$

  - L_2 regularization pulls coefficients toward 0

Regularization

Ridge Regression / Tikhonov regularization

- Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2$$

- Fit by solving $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

- Gradient update:

$\frac{\partial}{\partial\theta_0}J(\theta)$

$$\theta_0 \leftarrow \theta_0 - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)$$

$\frac{\partial}{\partial\theta_j}J(\theta)$

$$\theta_j \leftarrow \theta_j - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)} \underbrace{- \lambda\theta_j}_{\text{regularization}}$$

$$\theta_j \leftarrow \theta_j\left(1 - \alpha\lambda\right) - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)}$$

Regularization

Lasso Regression (Least Absolute Shrinkage and Selection Operator Regression)

- Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{d}|\theta_j|$$

- Fit by solving $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

- Gradient update:

$\frac{\partial}{\partial\theta_0}J(\theta)$

$$\theta_0 \leftarrow \theta_0 - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)$$

$\frac{\partial}{\partial\theta_j}J(\theta)$

$$\theta_j = \theta_j - \frac{\alpha}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)\boldsymbol{x}_j^{(i)} - \underbrace{\alpha\lambda\,\mathrm{sign}(\theta_j)}_{\text{regularization}}$$

$$\text{where} \quad \mathrm{sign}(\theta_i) = \begin{cases} -1 & \text{if } \theta_i < 0 \\ 0 & \text{if } \theta_i = 0 \\ +1 & \text{if } \theta_i > 0 \end{cases}$$

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