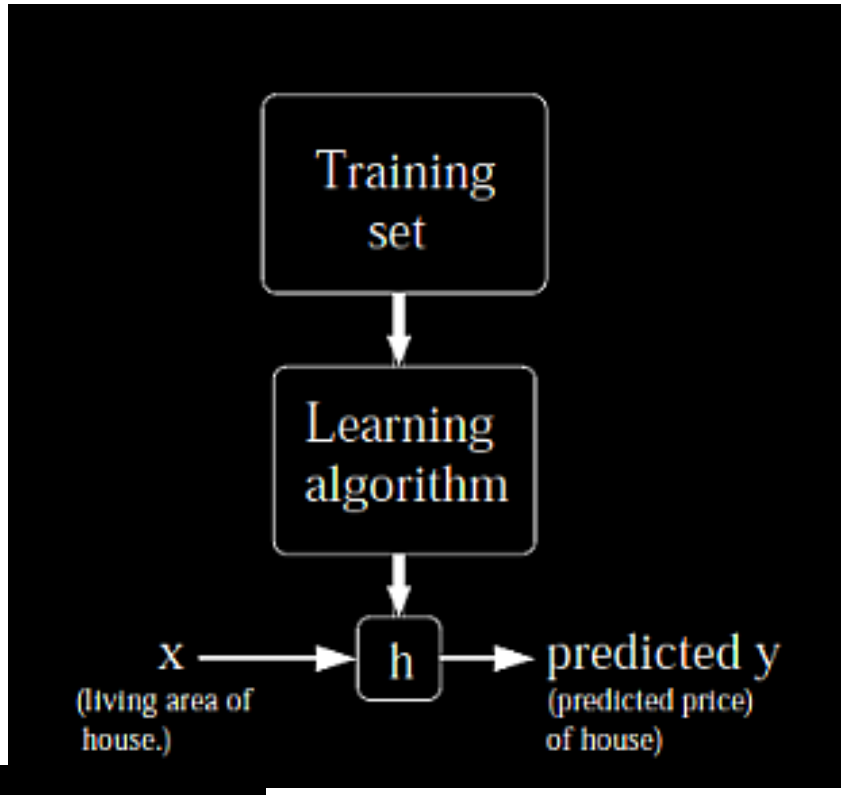
LOGISTIC REGRESSION

Bishop Sec. 4.3.2; Wiki;
Also,
Online Notes from Andrew Ng



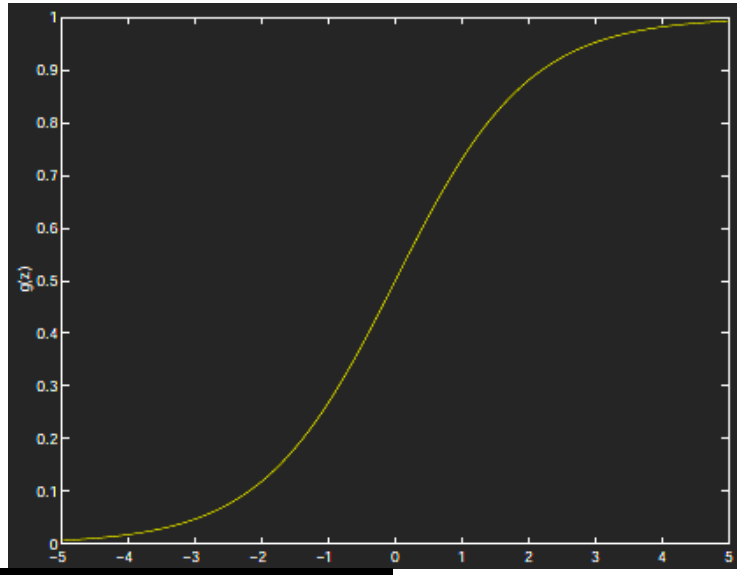
$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}},$$

where

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

is called the logistic function or the sigmoid function.

$$\theta^T x = \theta_0 + \sum_{j=1}^{n} \theta_j x_j$$



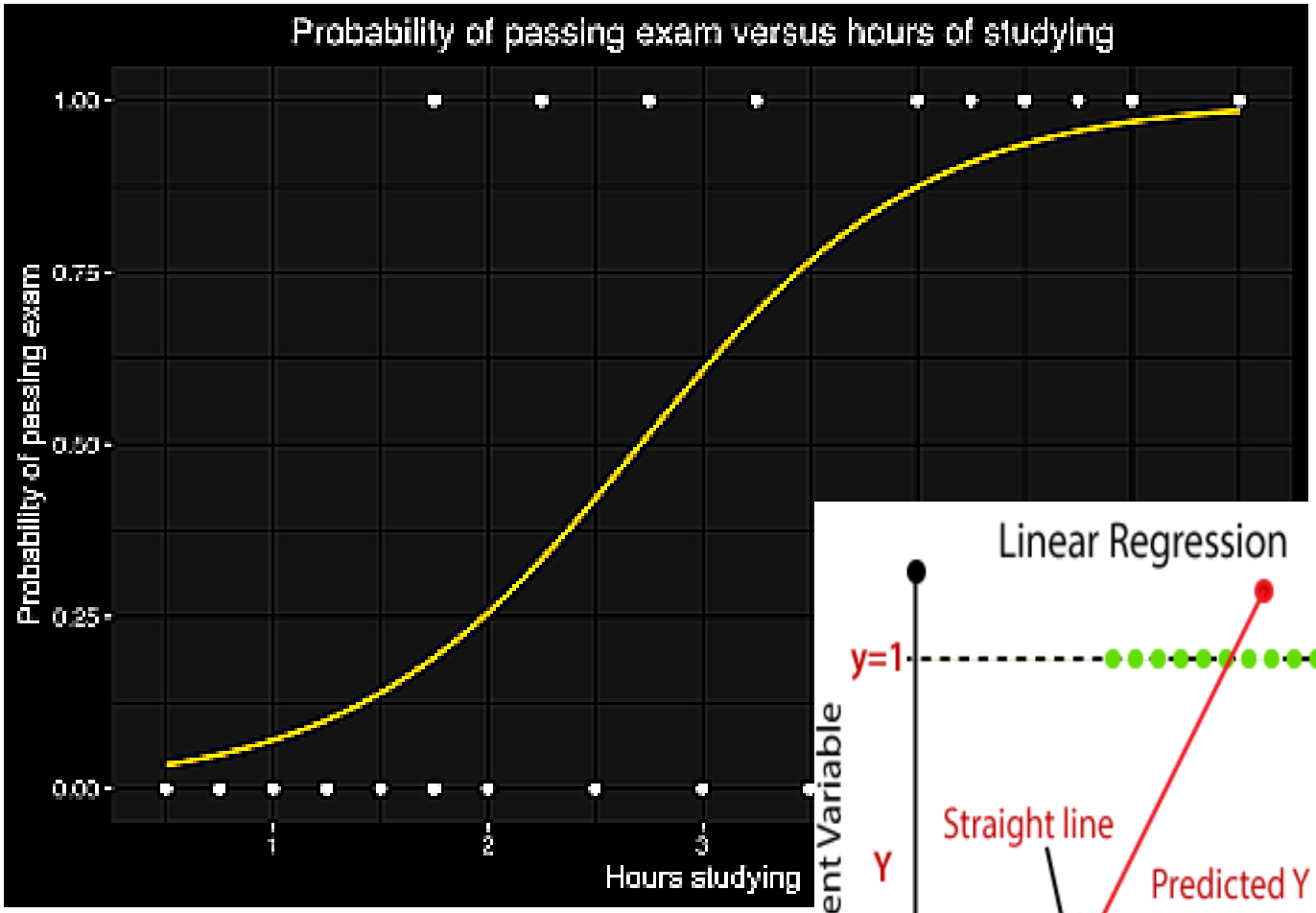
We define the cost function:

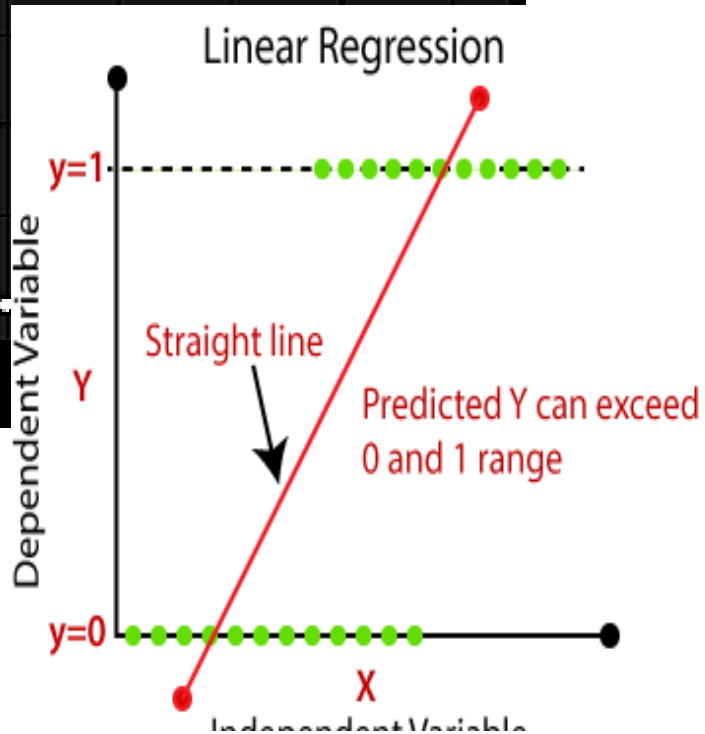
$$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

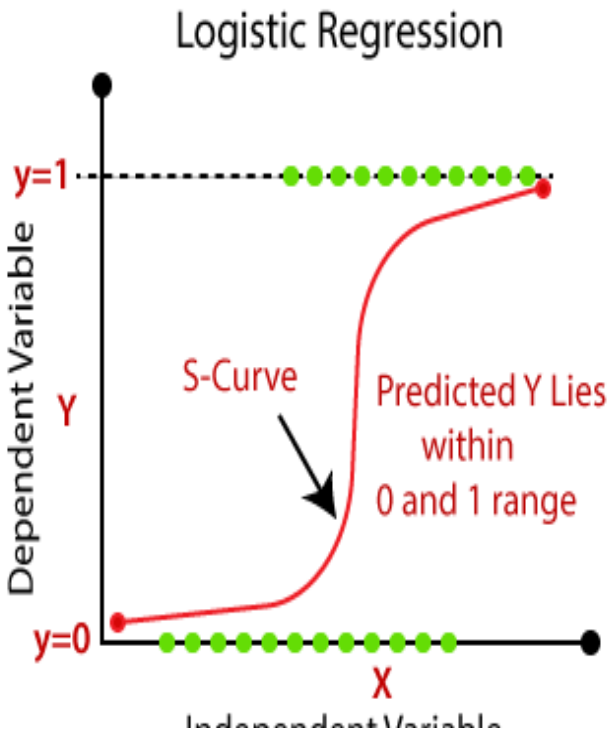
We want to choose θ so as to minimize $J(\theta)$. To do so, let's use a search algorithm that starts with some "initial guess" for θ, and that repeatedly changes θ to make $J(\theta)$ smaller, until hopefully we converge to a value of θ that minimizes $J(\theta)$. Specifically, let's consider the **gradient descent** algorithm, which starts with some initial θ, and repeatedly performs the update:

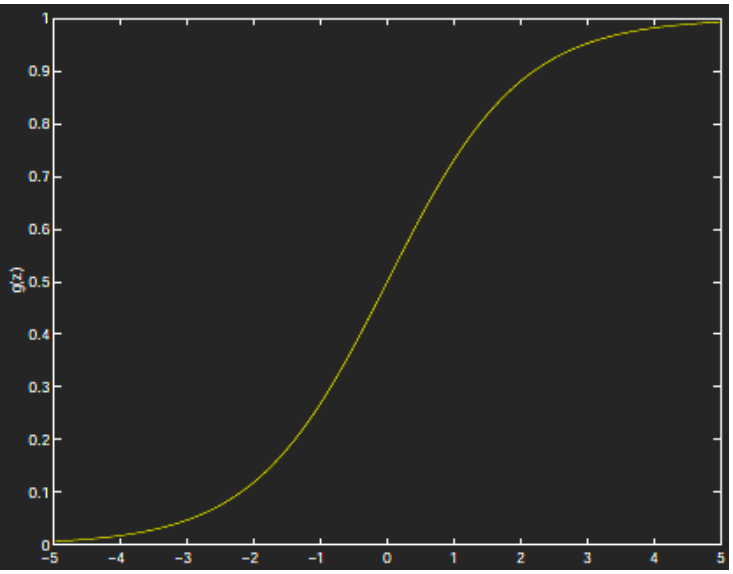
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta).$$

(This update is simultaneously performed for all values of $j = 0, \ldots, n$.) Here, α is called the **learning rate**. This is a very natural algorithm that repeatedly takes a step in the direction of steepest decrease of J.


Probability of passing exam versus hours of studying
1.00
0.75
0.50
0.25
0.00
Probability of passing exam
1
2
3
Hours studying


Linear Regression
y=1
y=0
Dependent Variable
Y
X
Straight line
Predicted Y can exceed
0 and 1 range


Logistic Regression
y=1
y=0
Dependent Variable
Y
X
S-Curve
Predicted Y Lies
within
0 and 1 range



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

where

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

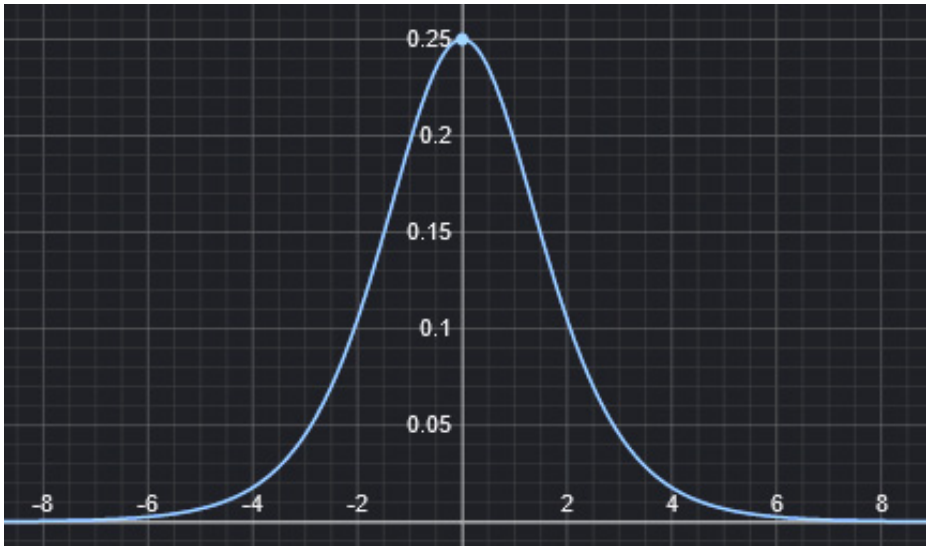
is called the logistic function or the sigmoid function.

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

$$=$$

$$=$$

$$=$$



← Draw the curve

Logistic Regression

$p(X) = \Pr(Y = 1|X)$

Logistic regression uses the form

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}.$$

($e \approx 2.71828$ is a mathematical constant [Euler's number.])
It is easy to see that no matter what values β_0, β_1 or X take, $p(X)$ will have values between 0 and 1.

A bit of rearrangement gives *(Solve it now)*

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \square$$

This monotone transformation is called the *log odds* or *logit* transformation of $p(X)$.

Definition of the inverse of the logistic function

We can now define the logit (log odds) function as the inverse $g = \sigma^{-1}$ of the standard logistic function. It is easy to see that it satisfies:

$$g(p(x)) = \sigma^{-1}(p(x)) = \operatorname{logit} p(x) = \ln\left(\frac{p(x)}{1-p(x)}\right) = \beta_0 + \beta_1 x,$$

and equivalently, after exponentiating both sides we have the odds:

$$\frac{p(x)}{1-p(x)} = e^{\beta_0+\beta_1 x}.$$

The odds ratio

For a continuous independent variable the odds ratio can be defined as:

$$\mathrm{OR} = \frac{\operatorname{odds}(x+1)}{\operatorname{odds}(x)} = \frac{\left(\frac{p(x+1)}{1-p(x+1)}\right)}{\left(\frac{p(x)}{1-p(x)}\right)} = \frac{e^{\beta_0+\beta_1(x+1)}}{e^{\beta_0+\beta_1 x}} = e^{\beta_1}$$

This exponential relationship provides an interpretation for β_1: The odds multiply by e^{β_1}

Maximum Likelihood

We use maximum likelihood to estimate the parameters.

$$\ell(\beta_0, \beta) = \prod_{i:y_i=1} p(x_i) \prod_{i:y_i=0} (1 - p(x_i)).$$

This *likelihood* gives the probability of the observed zeros and ones in the data. We pick β_0 and β_1 to maximize the likelihood of the observed data.

Logistic regression with several variables

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}{1 + e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}$$

So, given the logistic regression model, how do we fit θ for it?

Let us assume that

$$\begin{aligned} P(y = 1 \mid x; \theta) &= h_\theta(x) \\ P(y = 0 \mid x; \theta) &= 1 - h_\theta(x) \end{aligned}$$

Note that this can be written more compactly as

$$p(y \mid x; \theta) = (h_\theta(x))^y \, (1 - h_\theta(x))^{1-y}$$

Prob. model + MLE process

Assuming that the m training examples were generated independently, we can then write down the likelihood of the parameters as

$$\begin{aligned} L(\theta) &= p(\vec{y} \mid X; \theta) \\ &= \prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)}; \theta) \\ &= \prod_{i=1}^{m} \left(h_\theta(x^{(i)})\right)^{y^{(i)}} \left(1 - h_\theta(x^{(i)})\right)^{1-y^{(i)}} \end{aligned}$$

MLE

The negative log-likelihood for logistic regression is given by

Directly from Murphy; Sec. 8.3.1

$$\text{NLL}(\mathbf{w}) = -\sum_{i=1}^{N} \log[\mu_i^{\mathbb{I}(y_i=1)} \times (1-\mu_i)^{\mathbb{I}(y_i=0)}] \tag{8.2}$$

$$\boldsymbol{\mu = sigm(W^T x)}$$

$$= -\sum_{i=1}^{N} [y_i \log \mu_i + (1-y_i)\log(1-\mu_i)] \tag{8.3}$$

This is also called the **cross-entropy** error function (see Section 2.8.2).

$$NLL(\mathbf{w}) = \sum_{i=1}^{N} \log(1 + \exp(-\tilde{y}_i \mathbf{w}^T \mathbf{x}_i))$$

$$\mathbf{g} = \frac{d}{d\mathbf{w}} f(\mathbf{w}) = \sum_i (\mu_i - y_i)\mathbf{x}_i = \mathbf{X}^T(\mu - \mathbf{y}) \tag{8.5}$$

$$\mathbf{H} = \frac{d}{d\mathbf{w}} \mathbf{g}(\mathbf{w})^T = \sum_i (\nabla_{\mathbf{w}} \mu_i)\mathbf{x}_i^T = \sum_i \mu_i(1 - \mu_i)\mathbf{x}_i\mathbf{x}_i^T \tag{8.6}$$

$$= \mathbf{X}^T \mathbf{S} \mathbf{X} \tag{8.7}$$

where $\mathbf{S} \triangleq \text{diag}(\mu_i(1 - \mu_i))$. One can also show (Exercise 8.3) that $\mathbf{H}$ is positive definite.

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

it will be easier to maximiz

where

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

is called the logistic function or the sigmoid function.

$$\begin{aligned}
\ell(\theta) &= \log L(\theta) \\
&= \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1 - y^{(i)}) \log(1 - h(x^{(i)}))
\end{aligned}$$

one training example (x, y), and take de

$$\mathbf{g} = \frac{d}{d\mathbf{w}} f(\mathbf{w}) = \sum_i (\mu_i - y_i)\mathbf{x}_i$$

gradient ascent rule:

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} \ell(\theta) &= \left(y \frac{1}{g(\theta^T x)} - (1 - y) \frac{1}{1 - g(\theta^T x)} \right) \frac{\partial}{\partial \theta_j} g(\theta^T x) \\
&= \\
&= \\
&=
\end{aligned}$$

the stochastic gradient ascent rule

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

Same as LMS learning rule - except the non-linear sigmoid in "h".

Newton Raphson's method for maximizing $l(\theta)$

$$\theta := \theta - \frac{f(\theta)}{f'(\theta)}$$

$$\theta := \theta - \frac{\ell'(\theta)}{\ell''(\theta)}$$

$$\theta := \theta - H^{-1} \nabla_\theta \ell(\theta)$$

$$H_{ij} = \frac{\partial^2 \ell(\theta)}{\partial \theta_i \partial \theta_j}$$

As a generalized linear model

The particular model used by logistic regression, which distinguishes it from standard linear regression and from other types of regression analysis used for binary-valued outcomes, is the way the probability of a particular outcome is linked to the linear predictor function:

$$\operatorname{logit}(\mathbb{E}[Y_i \mid x_{1,i}, \ldots, x_{m,i}]) = \operatorname{logit}(p_i) = \ln\left(\frac{p_i}{1 - p_i}\right) = \beta_0 + \beta_1 x_{1,i} + \cdots + \beta_m x_{m,i}$$

Written using the more compact notation described above, this is:

$$\operatorname{logit}(\mathbb{E}[Y_i \mid \mathbf{X}_i]) = \operatorname{logit}(p_i) = \ln\left(\frac{p_i}{1 - p_i}\right) = \boldsymbol{\beta} \cdot \mathbf{X}_i$$

This formulation expresses logistic regression as a type of generalized linear model, which predicts variables with various types of probability distributions by fitting a linear predictor function of the above form to some sort of arbitrary transformation of the expected value of the variable.

Logistic regression with more than two classes

So far we have discussed logistic regression with two classes. It is easily generalized to more than two classes. One version (used in the R package `glmnet`) has the symmetric form

$$\Pr(Y = k|X) = \frac{e^{\beta_{0k}+\beta_{1k}X_1+\ldots+\beta_{pk}X_p}}{\sum_{\ell=1}^{K} e^{\beta_{0\ell}+\beta_{1\ell}X_1+\ldots+\beta_{p\ell}X_p}}$$

Here there is a linear function for *each* class.

Multiclass logistic regression is also referred to as *multinomial regression*.

Consider a general classification problem, in which the response variable y can take on any one of k values, so y ∈ {1, 2, . . . , k}.

To parameterize a multinomial over k possible outcomes, one could use k parameters $\phi_1, \ldots, \phi_k$ specifying the probability of each of the outcomes.

$$\text{let } \phi_k = 1 - \sum_{i=1}^{k-1} \phi_i$$

(for i = 1, . . . , k)

$$\eta_i = \log \frac{\phi_i}{\phi_k}$$

response function

$$\phi_i = \frac{e^{\eta_i}}{\sum_{j=1}^{k} e^{\eta_j}}$$

response function

$$\phi_i = \frac{e^{\eta_i}}{\sum_{j=1}^{k} e^{\eta_j}}$$

This function mapping from the η's to the ϕ's is called the softmax function

$$\eta_i = \theta_i^T x \text{ (for } i = 1, \ldots, k-1)$$

where $\theta_1, \ldots, \theta_{k-1} \in \mathbb{R}^{n+1}$ are the parameters of our model

The conditional distribution of y given x is :

$$
\begin{aligned}
p(y = i|x;\theta) &= \phi_i \\
&= \frac{e^{\eta_i}}{\sum_{j=1}^{k} e^{\eta_j}} \\
&= \frac{e^{\theta_i^T x}}{\sum_{j=1}^{k} e^{\theta_j^T x}}
\end{aligned}
$$

This model, which applies to classification problems where y ∈ {1, . . . , k}, is called softmax regression.
It is a generalization of logistic regression.

If we have a training set of m examples {(x(i), y(i)); i = 1, . . . ,m} and would like to learn the parameters θ_i of this model, write down the log-likelihood, as:

$$\begin{aligned}\ell(\theta) &= \sum_{i=1}^{m} \log p(y^{(i)}|x^{(i)};\theta) \\ &= \sum_{i=1}^{m} \log \prod_{l=1}^{k} \left(\frac{e^{\theta_l^T x^{(i)}}}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \right)^{1\{y^{(i)}=l\}}\end{aligned}$$

LOGISTIC REGRESSION (to rewind/Altn symbols)

- The posterior probability of class C_1 can be written as a logistic sigmoid acting on a linear function of the feature vector ϕ so that

$$p(\mathcal{C}_1|\boldsymbol{\phi}) = y(\boldsymbol{\phi}) = \sigma\left(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}\right)$$

  with $p(\mathcal{C}_2|\boldsymbol{\phi}) = 1 - p(\mathcal{C}_1|\boldsymbol{\phi})$.

- Here, $\sigma(\cdot)$ is the *logistic sigmoid* function defined by

$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$

- In the terminology of statistics, this model is known as *logistic regression*, although it should be emphasized that this is a model for classification rather than regression.

LOGISTIC REGRESSION

- For an *M*-dimensional feature space ϕ, this model has *M* adjustable parameters.
- By contrast, if we had fitted Gaussian class conditional densities using maximum likelihood, we would have used 2*M* parameters for the means and $M(M + 1)/2$ parameters for the (shared) covariance matrix.
- Together with the class prior $p(C_1)$, this gives a total of $M(M+5)/2+1$ parameters, which grows quadratically with *M*, in contrast to the linear dependence on *M* of the number of parameters in logistic regression.
- For large values of *M*, there is a clear advantage in working with the logistic regression model directly.

LOGISTIC REGRESSION

- We now use maximum likelihood to determine the parameters of the logistic regression model.
- To do this, we shall make use of the derivative of the logistic sigmoid function, which can conveniently be expressed in terms of the sigmoid function itself

$$\frac{d\sigma}{da} = \sigma(1-\sigma).$$

- For a data set $\{\phi_n, t_n\}$, where $t_n \in \{0, 1\}$ and $\phi_n = \phi(\mathbf{x}_n)$, with $n = 1, \ldots, N$, the likelihood function can be written

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} \{1 - y_n\}^{1-t_n}$$

LOGISTIC REGRESSION

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} \{1 - y_n\}^{1-t_n}$$

Here $\mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$ and $y_n = p(\mathcal{C}_1|\boldsymbol{\phi}_n)$

- We can define an error function by taking the negative logarithm of the likelihood, which gives the *cross entropy*

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

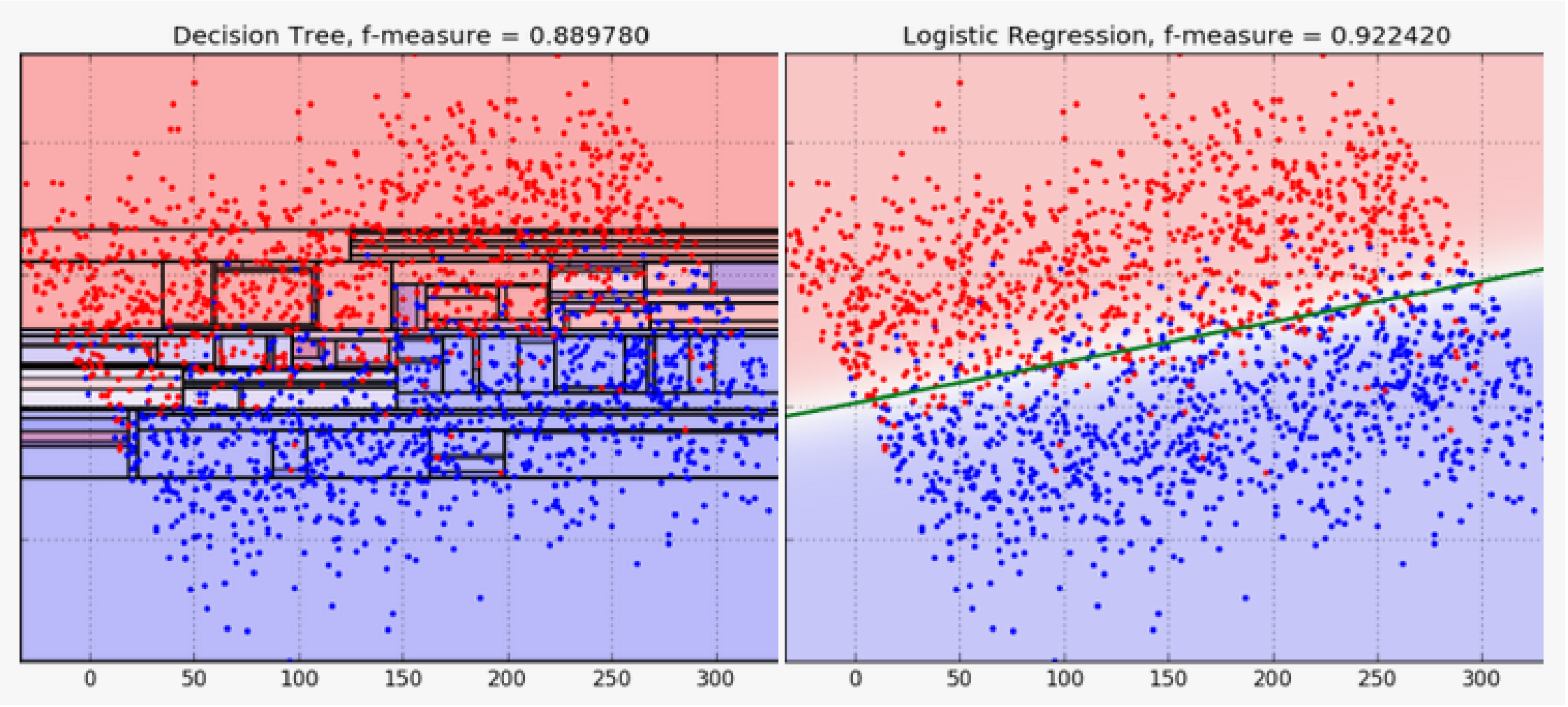
where, $y_n = \sigma(a_n)$ and $a_n = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}_n$

LOGISTIC REGRESSION

- Taking the gradient of the error function with respect to **w**, we obtain

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\boldsymbol{\phi}_n$$

- where we have made use of the derivative of sigmoid.
- In particular, the contribution to the gradient from data point *n* is given by the 'error' $y_n - t_n$ between the target value and the prediction of the model, times the basis function vector $\boldsymbol{\phi}_n$.


Decision Tree, f-measure = 0.889780
Logistic Regression, f-measure = 0.922420
0
50
100
150
200
250
300
0
50
100
150
200
250
300

----- XXXX -------