MSc in AI
NCSR Demokritos - University of Piraeus

Course: **Machine Learning**

Lesson 3
Logistic Regression

Theodoros Giannakopoulos

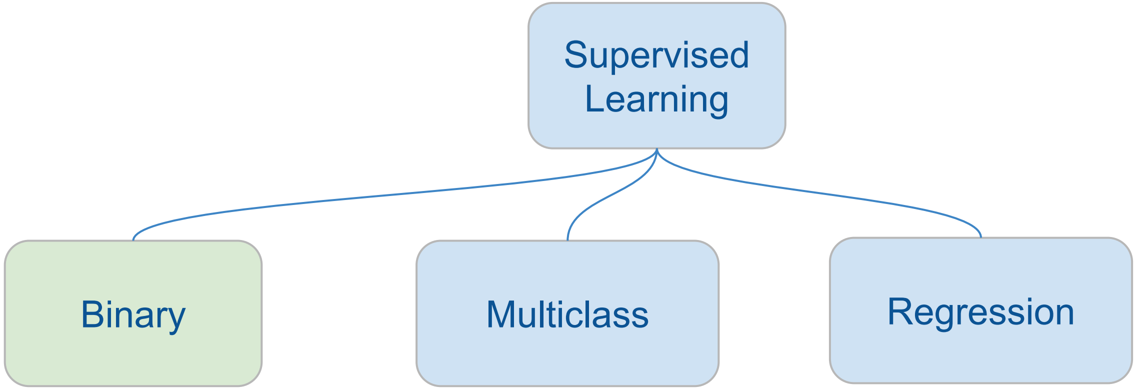
Recap: Supervised learning

- Given data and correct output, try to find a relationship between data and output

$$\{(x_i, y_i)\}_{i=1}^{N}$$

- N labelled examples or instances or feature vectors
- x_i feature vector $\in \mathbb{R}^D$
- D: dimensionality
- $x_i(j)$ j-th feature, $j=1, \ldots, D$
- y_i label $\in \{w1, w2, \ldots, wc\}$ set of **classes** OR $\mathbb{R}$ (regression)

Recap: Types of supervised learning

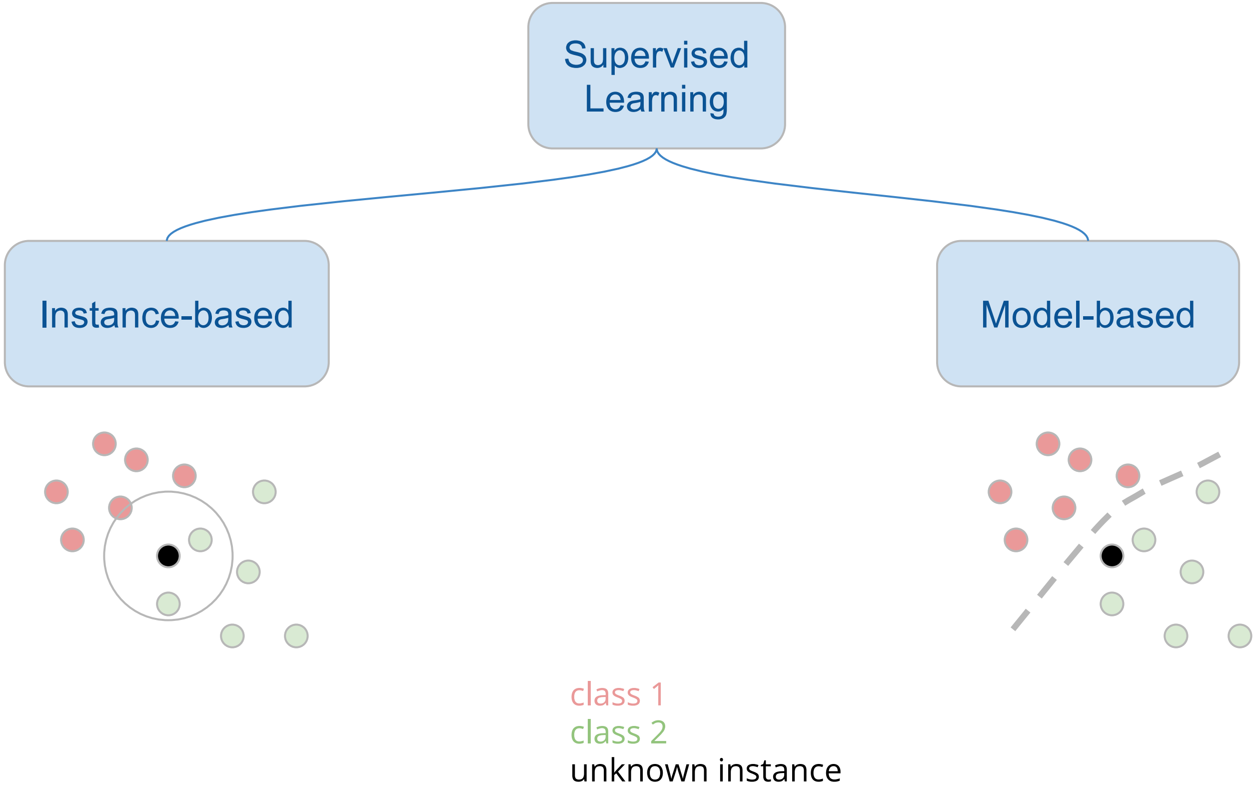


sound is either music or speech?

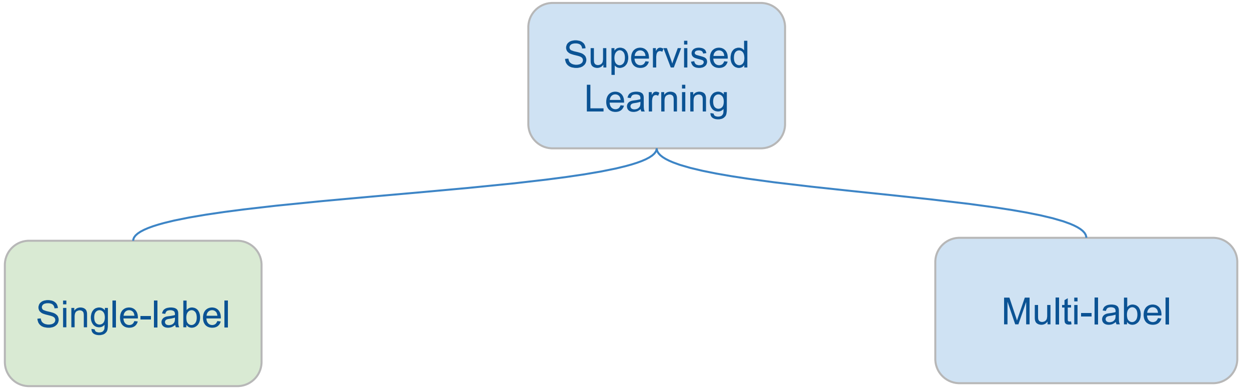
song is jazz, pop, electro or rock?

predict the value of this product next year

Recap: Types of supervised learning



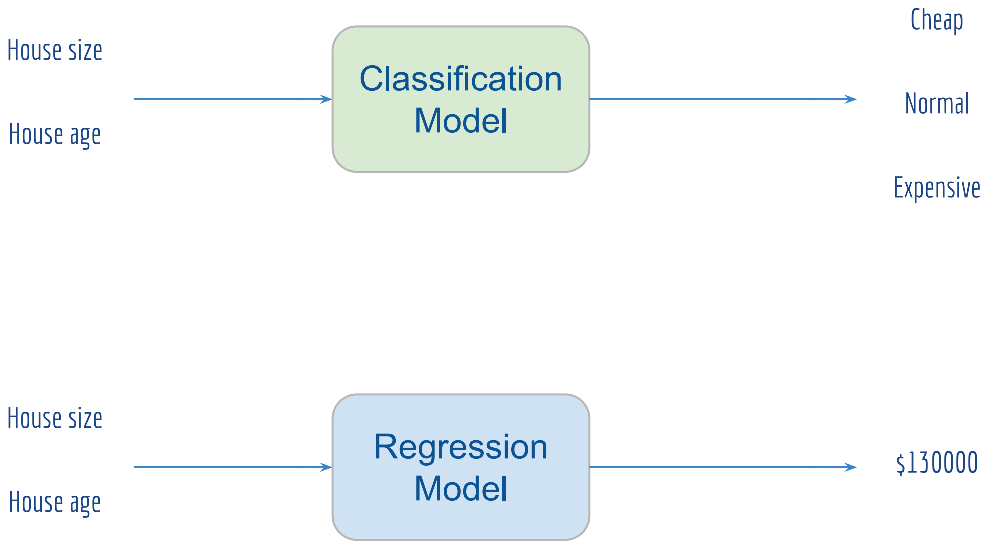
Recap: Types of supervised learning



song1: jazz
song2: jazz
song3: electronic
song4: ...

song1: electro-swing, hip-hop
song2: acid-jazz, funk
song3: detroit-techno
song4: ...

Recap: Classification Vs Regression



Recap: Linear Regression

$$f_{\boldsymbol{w},b}(\boldsymbol{x}) = \boldsymbol{w}\boldsymbol{x} + b$$

$$J(\boldsymbol{w}, b) = \frac{1}{2N}\sum_{i=0}^{N}(f_{\boldsymbol{w},b}(x_i) - y_i)^2$$

Cost function: measures the error between true and predicted values

Goal:

minimize J to find w and b

Loss function: a measure of penalty for misclassification of each example i

In linear regression, cost function is the average loss (also called empirical risk)

Recap: Linear Regression: how is J minimized? (GD for 2 params)

$$J(w_1, b) = \frac{1}{2N}\sum_{i=0}^{N}(f_{w_1,b}(x_i) - y_i)^2 \qquad f_{w_1,b} = w_1 x + b$$

Gradient Descent:

- Select a random value for w1 and b
- Until convergence (or for a max number of epochs):

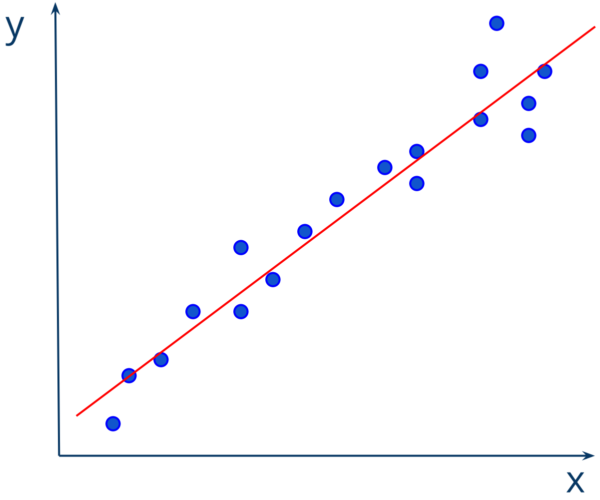
$$w_1 := w_1 - \alpha\frac{\partial J(w_1, b))}{\partial w_1} = w_1 - \alpha\frac{1}{N}\sum_{i=0}^{N}(f_{w_1,b}(x) - y)x$$

$$b := b - \alpha\frac{\partial J(w_1, b))}{\partial b} = b - \alpha\frac{1}{N}\sum_{i=0}^{N}(f_{w_1,b}(x) - y)$$

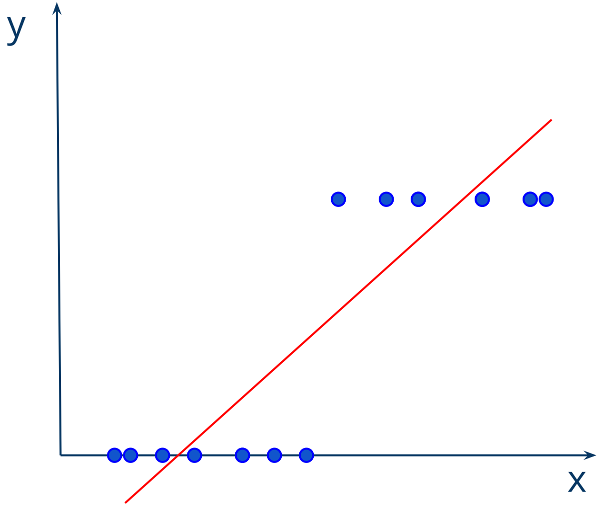
Why? Just forget Σ and:

$$\begin{aligned}\frac{\partial}{\partial\theta_j}J(\theta) &= \frac{\partial}{\partial\theta_j}\frac{1}{2}(f_{\theta,b}(x) - y)^2 = \\ &= 2\frac{1}{2}(f_{\theta,b}(x) - y)\frac{\partial}{\partial\theta_j}(f_{\theta,b}(x) - y) = \\ &= (f_{\theta,b}(x) - y)\frac{\partial}{\partial\theta_j}(f_{\theta,b}(x) - y) = \\ &= (f_{\theta,b}(x) - y)\frac{\partial}{\partial\theta_j}(\textstyle\sum_i \theta_i x_i - y) = \\ &= (f_{\theta,b}(x) - y)x_j\end{aligned}$$

Linear Regression for Binary Classification?



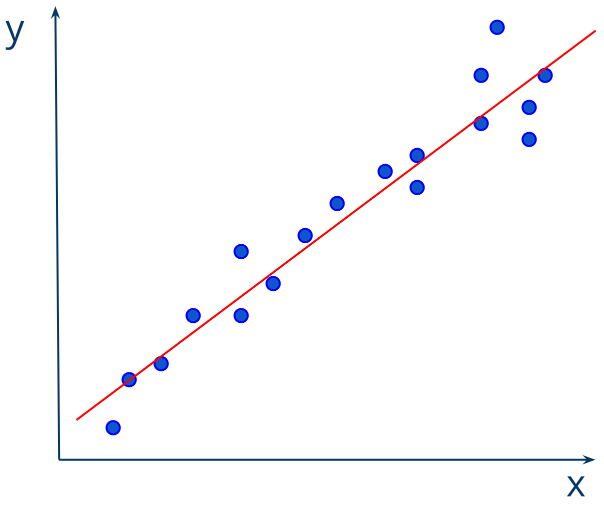
Regression: $y \in \mathbb{R}$



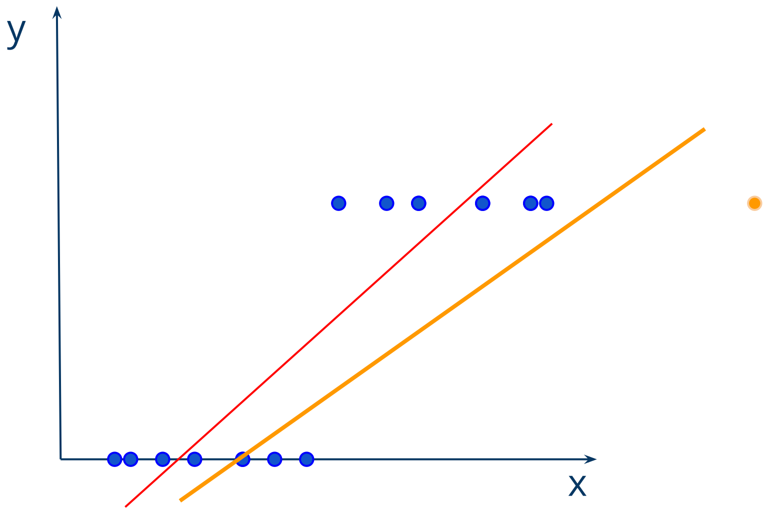
(Binary) Classification: **$y \in \{0, 1\}$**

So can we achieve classification through linear regression and then threshold f(x) with T = 0.5 ?
(if $f(x) > 0.5 \longrightarrow f(x) = 1$ else $f(x) = 0$

Linear Regression for Binary Classification?



Regression: $y \in \mathbb{R}$



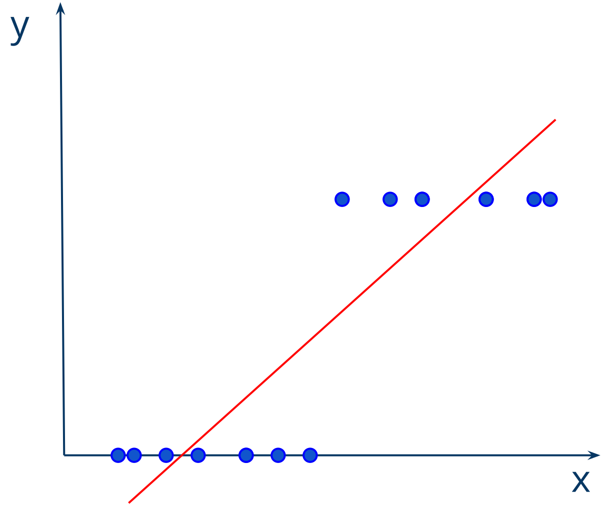
(Binary) Classification: **$y \in \{0, 1\}$**

So can we achieve classification through linear regression and then threshold f(x) with T = 0.5 ?
(if $f(x) > 0.5 \longrightarrow f(x) = 1$ else $f(x) = 0$

NO!

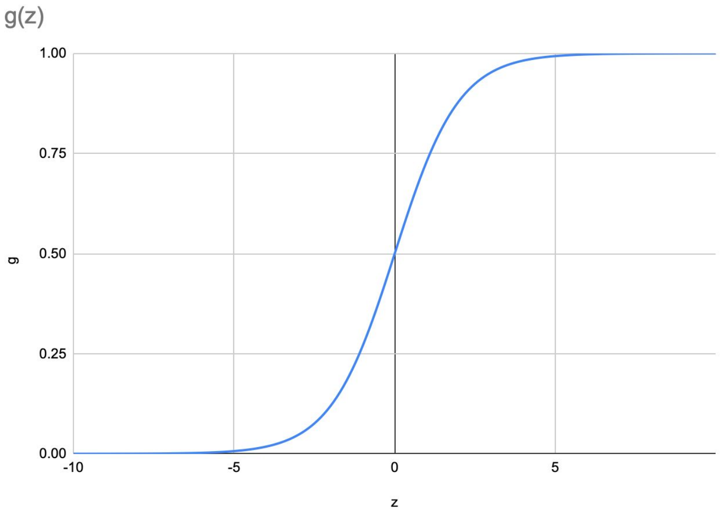
Logistic Regression for Binary Classification

- **$y \in \{0, 1\}$**
- we want our model f(x) to be in [0, 1]
- HOW?



Logistic Binary Classification

- **$y \in \{0, 1\}$**
- we want our model f(x) to be in [0, 1]
- HOW: define **sigmoid** or **logistic** function $g(z) = \dfrac{1}{1 + e^{-z}}$



Logistic Regression: the logistic function

- **$y \in \{0, 1\}$**
- we want our model f(x) to be in [0, 1]
- So instead of a linear relationship ($f=w^T*x$) we use the sigmoid or logistic that guaranties [0, 1] range:

$$f_{\mathbf{w}}(\mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w}^T \mathbf{x}}}$$

Logistic Regression: class probability given x and w

- $y \in \{0, 1\}$
- probability that class is 1 given x and the classifier's param w $P(y=1|\mathbf{x},\mathbf{w}) = f_{\mathbf{w}}(\mathbf{x})$
- probability that class is 0 given x and the classifier's param w $P(y=0|\mathbf{x},\mathbf{w}) = 1 - f_{\mathbf{w}}(\mathbf{x})$
- "Compress" the two equations above to one:

$$P(y|\mathbf{x},\mathbf{w}) = f_{\mathbf{w}}(\mathbf{x})^y (1 - f_{\mathbf{w}}(\mathbf{x}))^{1-y}$$

- Goal: Maximize conditional likelihood P(y|x, w)
- Interval: what is Maximum Likelihood?

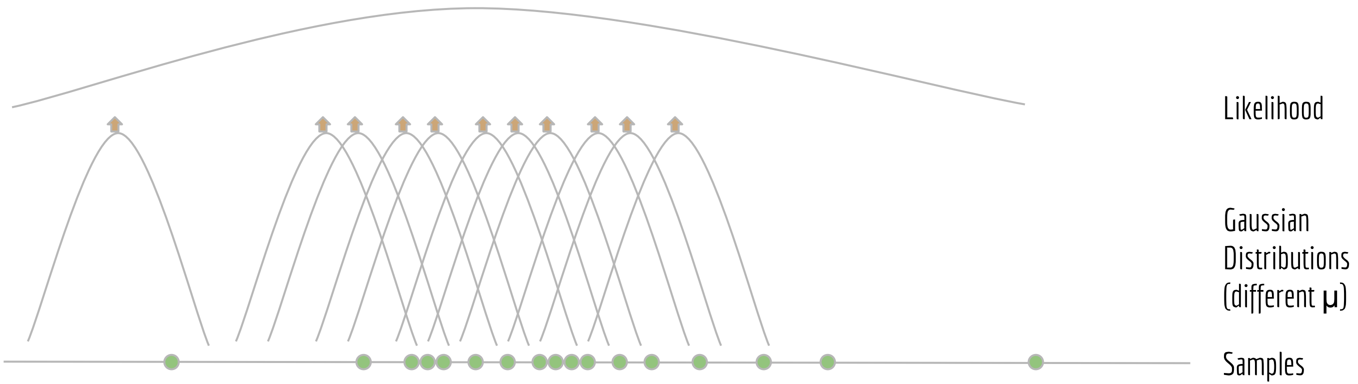
Maximum Likelihood Estimation: General

- Problem: fit a distribution to your data
- Why? Easier to work with distributions rather than raw data “values”
- Let ϴ be a parameter
- Let x1, x2, …, xN be random samples from P(x|ϴ)
- ML Estimation answers the question: find the parameter(s) of a distribution that “fit” my data OR:
- What is the most **likely** value of ϴ given my data x…?
- Likelihood function:

$$L(\theta) = P(x_1, \ldots, x_N|\theta) = \prod_{i=1}^{N} P(x_i|\theta)$$

- MLE: maximize L(ϴ) or (for practical reasons) its log

Maximum Likelihood Estimation: General



$$L(\theta) = P(x_1, \ldots, x_N|\theta) = \prod_{i=1}^{N} P(x_i|\theta)$$

Logistic Regression: (log) conditional likelihood

- So we have expressed our “logistic model” according to $f_{\mathbf{w}}(\mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w}^T \mathbf{x}}}$
- And the class probability given the feature and the model param w is

$$P(y|\mathbf{x}, \mathbf{w}) = f_{\mathbf{w}}(\mathbf{x})^y (1 - f_{\mathbf{w}}(\mathbf{x}))^{1-y}$$

- In LR we optimize the **likelihood** of our training data according to our model = “how likely the observation is according to our model”

$$L(\mathbf{w}) = \prod_{i=1}^{N} p(y|x, w) = \prod_{i=1}^{N} f_{\mathbf{w}}(\mathbf{x}_i)^{y_i} (1 - f_{\mathbf{w}}(\mathbf{x}_i))^{1-y_i}$$

- “f(x) when y=1 and (1-f) otherwise”: $f_{\mathbf{w}}(\mathbf{x}_i)^{y_i} (1 - f_{\mathbf{w}}(\mathbf{x}_i))^{1-y_i}$
- Observations xi, yi are independent $\longrightarrow$ likelihood of N observations = product of N likelihoods
- More convenient to use log-likelihood:

$$\ell(\mathbf{w}) = ln(L(\mathbf{w})) = \sum_{i=1}^{N} y_i ln f_{\mathbf{w}}(\mathbf{x}_i) + (1 - y_i) ln(1 - f_{\mathbf{w}}(\mathbf{x}_i))$$

Logistic Regression: log-likelihood

- Maximize log-likelihood

$$\ell(\mathbf{w}) = ln(L(\mathbf{w})) = \sum_{i=1}^{N} y_i ln f_{\mathbf{w}}(\mathbf{x}_i) + (1 - y_i) ln(1 - f_{\mathbf{w}}(\mathbf{x}_i))$$

- MLE is the approach we follow to estimate the parameters w of the model. HOW?
- **Gradient Descent** used again as a maximization algorithm (actually gradient ascent as we now focus on maximizing)
- No closed form like linear regression

Logistic Regression: maximize log-likelihood

- Goal: maximize log-likelihood:

$$\ell(\mathbf{w}) = ln(L(\mathbf{w})) = \sum_{i=1}^{N} y_i ln f_{\mathbf{w}}(\mathbf{x}_i) + (1 - y_i) ln(1 - f_{\mathbf{w}}(\mathbf{x}_i))$$

- Solution:

$$w_j := w_j + a \frac{\partial}{\partial w_j} \ell(\mathbf{w})$$

- Or:

$$\boxed{w_j := w_j + a \sum_{i=1}^{N} (y^{(i)} - f_{\mathbf{w}}(\mathbf{x}^{(i)}))) x_j^{(i)})}$$

Logistic Regression: maximize log-likelihood

- Goal: maximize log-likelihood:

$$\ell(\mathbf{w}) = ln(L(\mathbf{w})) = \sum_{i=1}^{N} y_i ln f_{\mathbf{w}}(\mathbf{x}_i) + (1 - y_i) ln(1 - f_{\mathbf{w}}(\mathbf{x}_i))$$

- Solution:

$$w_j := w_j + a \frac{\partial}{\partial w_j} \ell(\mathbf{w})$$

- Or:

$$w_j := w_j + a \sum_{i=1}^{N} (y^{(i)} - f_{\mathbf{w}}(\mathbf{x}^{(i)}))) x_j^{(i))}$$

The reason we use the logistic function in particular to map x to [0, 1] is that it guarantees that this has a global maximum

Logistic Regression: maximize log-likelihood

- Goal: maximize log-likelihood:

$$\ell(\mathbf{w}) = ln(L(\mathbf{w})) = \sum_{i=1}^{N} y_i ln f_{\mathbf{w}}(\mathbf{x}_i) + (1 - y_i) ln(1 - f_{\mathbf{w}}(\mathbf{x}_i))$$

- Solution:

$$w_j := w_j + a \frac{\partial}{\partial w_j} \ell(\mathbf{w})$$

- Or:

$$w_j := w_j + a \sum_{i=1}^{N} (y^{(i)} - f_{\mathbf{w}}(\mathbf{x}^{(i)})) x_j^{(i)})$$

- In linear regression:
  - goal: minimize square error
  - and cost function J
- Solution:

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

- Or:

$$w_j := w_j - a \sum_{i=1}^{N} (f_{\mathbf{w}}(\mathbf{x}^{(i)}) - y^{(i)}) x_j^{(i)})$$

So are they the same?

Logistic Regression: maximize log-likelihood

- Goal: maximize log-likelihood:

$$\ell(\mathbf{w}) = ln(L(\mathbf{w})) = \sum_{i=1}^{N} y_i ln f_{\mathbf{w}}(\mathbf{x}_i) + (1-y_i) ln(1 - f_{\mathbf{w}}(\mathbf{x}_i))$$

- Solution:

$$w_j := w_j + a \frac{\partial}{\partial w_j} \ell(\mathbf{w})$$

- Or:

$$w_j := w_j + a \sum_{i=1}^{N} (y^{(i)} - f_{\mathbf{w}}(\mathbf{x}^{(i)})) x_j^{(i)}$$

- In linear regression:
  - goal: minimize square error
  - and cost function J
- Solution:

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

- Or:

$$w_j := w_j - a \sum_{i=1}^{N} (f_{\mathbf{w}}(\mathbf{x}^{(i)}) - y^{(i)}) x_j^{(i)}$$

The difference lies in the way f is defined

Logistic Regression: Overview & Example

- Goal: maximize log-likelihood:

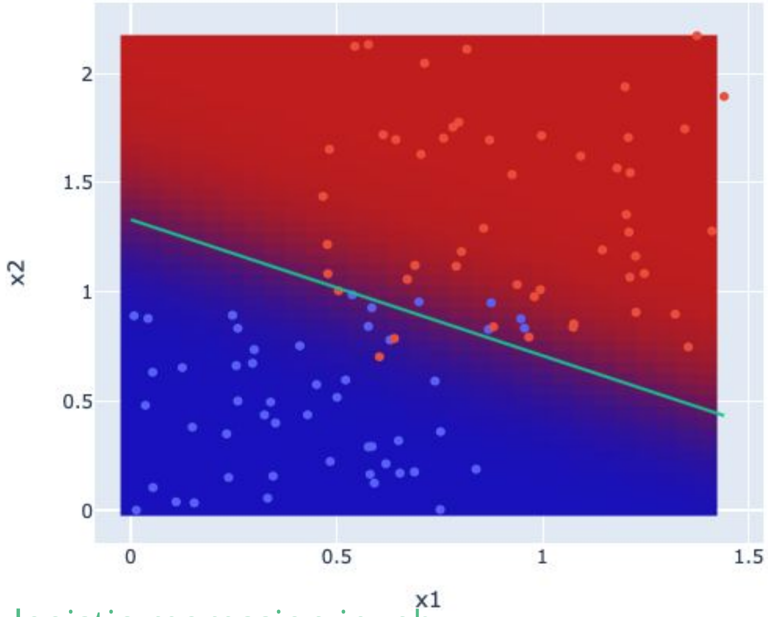
$$\ell(\mathbf{w}) = ln(L(\mathbf{w})) = \sum_{i=1}^{N} y_i ln f_{\mathbf{w}}(\mathbf{x}_i) + (1 - y_i) ln(1 - f_{\mathbf{w}}(\mathbf{x}_i))$$

$$f_{\mathbf{w}}(\mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w}^T\mathbf{x}}}$$

- Solution:

$$w_j := w_j + a\frac{\partial}{\partial w_j}\ell(\mathbf{w})$$

$$w_j := w_j + a\sum_{i=1}^{N}(y^{(i)} - f_{\mathbf{w}}(\mathbf{x}^{(i)}))x_j^{(i)}$$



* code available in
https://github.com/tyiannak/ml-python/blob/main/notebooks/2-logistic-regression.ipynb