# LINEAR MODEL AS CLASSIFIER

Dr. Hilman F. Pardede

*Research Center for Informatics*
Indonesian Institute of Sciences

The materials are compiled from the following resources:

- https://github.com/joaquinvanschoren/ML-course
- https://www.cse.iitk.ac.in/users/piyush/courses/ml_autumn16/ML.html
- http://sli.ics.uci.edu/Classes/2015W-273a

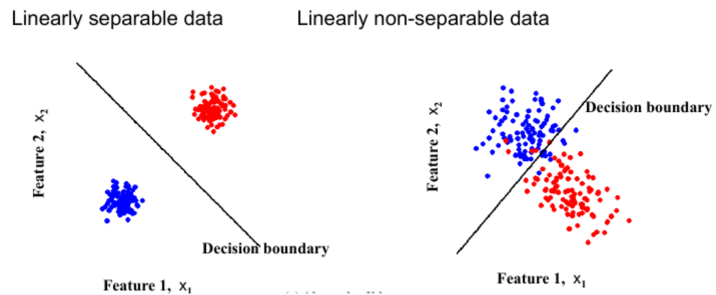
# LINEAR MODELS

Aims to find a (hyper)plane that separates the examples of each class.

For binary classification (2 classes), we aim to fit the following function:

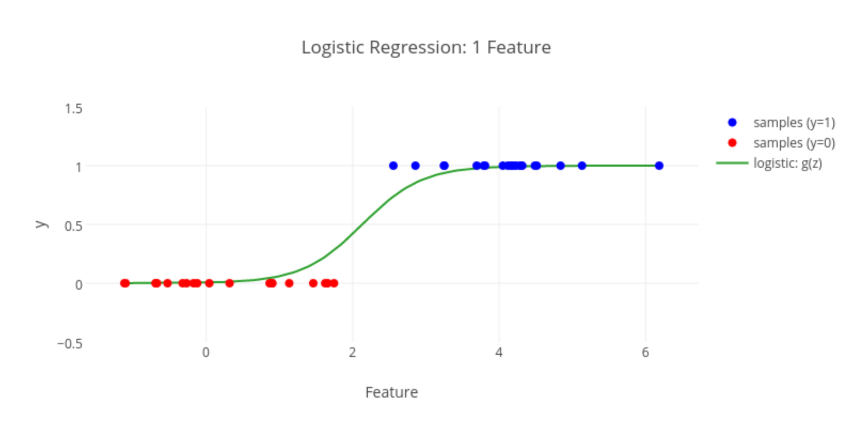
$\hat{y} = w_0 * x_0 + w_1 * x_1 + ... + w_p * x_p + b > 0$

When $\hat{y} < 0$, predict class -1, otherwise predict class +1



- There are many algorithms for learning linear classification models, differing in:
  - Loss function: evaluate how well the linear model fits the training data
  - Regularization techniques
- Most common techniques:
  - Logistic regression: `sklearn.linear_model.LogisticRegression`
  - Linear Support Vector Machine: `sklearn.svm.LinearSVC`

Fits a logistic regression curve/surface to the data



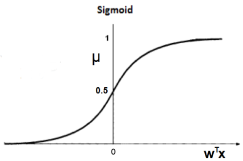
- A model for doing probabilistic binary classification
- Predicts label probabilities rather than a hard value of the label

$$p(y_n = 1 | \mathbf{x}_n, \mathbf{w}) = \mu_n \tag{1}$$

$$p(y_n = 0 | \mathbf{x}_n, \mathbf{w}) = 1 - \mu_n \tag{2}$$

- The model's prediction is a probability defined using the sigmoid function

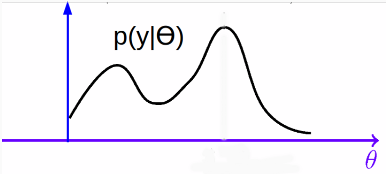
$$f(\mathbf{x}) = \mu_n = \sigma(\mathbf{w}^\top \mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^\top \mathbf{x})} \tag{3}$$



- The sigmoid first computes a real-valued "score" $\mathbf{w}^\top \mathbf{x} = \sum_{d=1}^{D} w_d x_d$ and "squashes" it between to turn it into a probability score (0,1)
- Model parameter is the unknown $\mathbf{w}$. Need to learn it from training data.
- Maximum likelihood is one of the method to find $\mathbf{w}$.
    - Similar to gradient descent
    - It can be done using iterative re-weighted least squares
    - Other optimization methods: Maximum A Posteriori

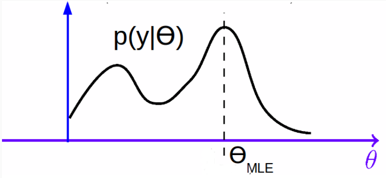
- given $\mathbf{y} = \{y_1, y_2, ..., y_N\}$, then the probability of observing $\mathbf{y}$, given features, assuming i.i.d:
- $p(\mathbf{y}|\theta) = p(y_1, y_2, ..., y_N|\theta) = \prod_{n=1}^{N} p(y_n|\theta)$
- $p(\mathbf{y}|\theta)$ also called the likelihood

- given $\mathbf{y} = \{y_1, y_2, ..., y_N\}$, then the probability of observing $\mathbf{y}$, given features, assuming i.i.d:
- $p(\mathbf{y}|\theta) = p(y_1, y_2, ..., y_N|\theta) = \prod_{n=1}^{N} p(y_n|\theta)$
- $p(\mathbf{y}|\theta)$ also called the likelihood

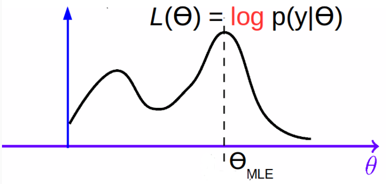



- How do we estimate the "best" model parameters ?

- given $\mathbf{y} = \{y_1, y_2, ..., y_N\}$, then the probability of observing $\mathbf{y}$, given features, assuming i.i.d:
- $p(\mathbf{y}|\theta) = p(y_1, y_2, ..., y_N|\theta) = \prod_{n=1}^{N} p(y_n|\theta)$
- $p(\mathbf{y}|\theta)$ also called the likelihood




- How do we estimate the "best" model parameters ?
- One option: Find value of $\theta$ that makes observed data most probable: Maximize the likelihood $p(\mathbf{y}|\theta)$ w.r.t. $\theta \Rightarrow$ **Maximum Likelihood Estimation**

- We doing MLE, we typically maximize log-likelihood instead of the likelihood, which is easier (doesn't affect the estimation because log is monotonic) 

- Log-likelihood:

$$\mathfrak{L}(\mathbf{y}|\theta) = \log p(\mathbf{y}|\theta) = \log \prod_{n=1}^{N} p(y_n|\theta) = \sum_{n=1}^{N} p(y_n|\theta) \log p(y_n|\theta) \quad (4)$$

- Maximum Likelihood Estimation (MLE)