Hochschule Karlsruhe
University of
Applied Sciences

Fakultät für
Elektro- und
Informationstechnik

Classification (2)

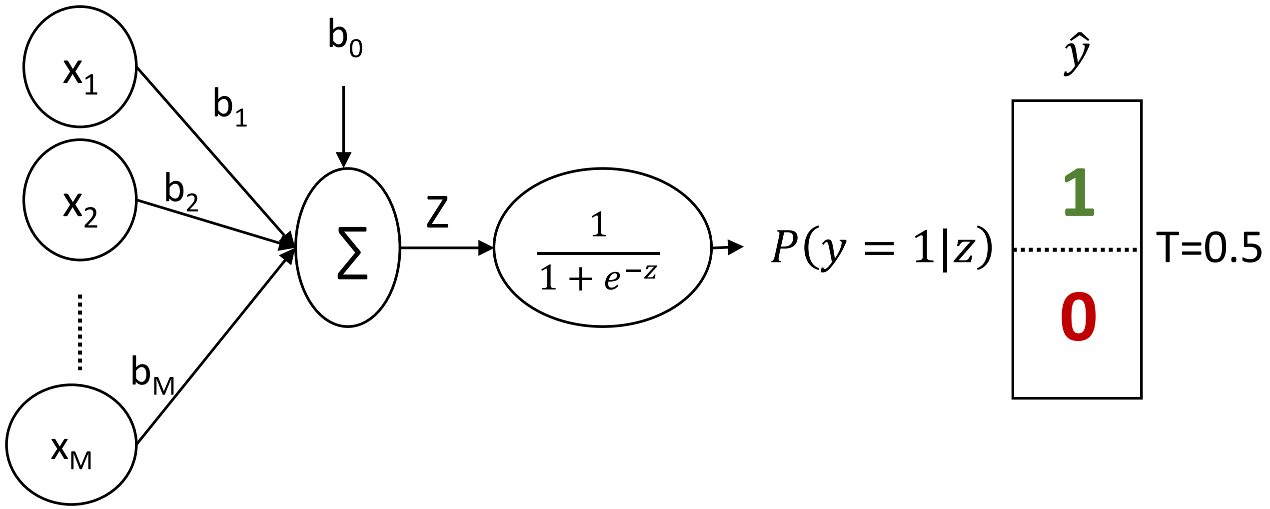
Source: DALL.E

Kawther Aboalam

Classification

Multiclass Classification

+ Logistic regression for binary classification can be illustrated by the shown block diagram
+ Logistic regression can be extended to solve multiclass problems, which involve more than two distinct classes.
+ $y \in \{1,2,\dots,K\}$ where K is the number of classes
+ Common Approaches:
    1. One-vs-Rest (One-vs-All)
    2. Softmax Regression (Multinominal Logistic Regression)

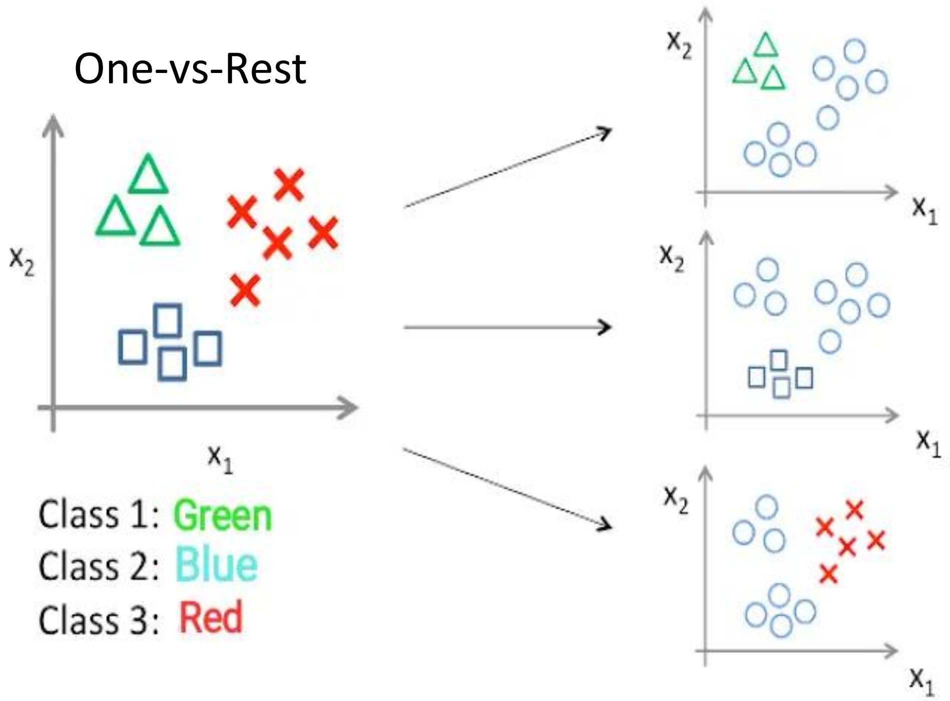


Classification

Multiclass Classification – One-vs-Rest (One-vs-All)

+ In general:
    + One-vs-Rest can be implemented by splitting up the multi-class classification problem into multiple binary classifier models.
    + Example: Suppose you have classes A, B, and C. We will build one model for each class:
        + Model 1: A or not A
        + Model 2: B or not B
        + Model 3: C or not C

+ It is similar to the concept of one-hot encoder



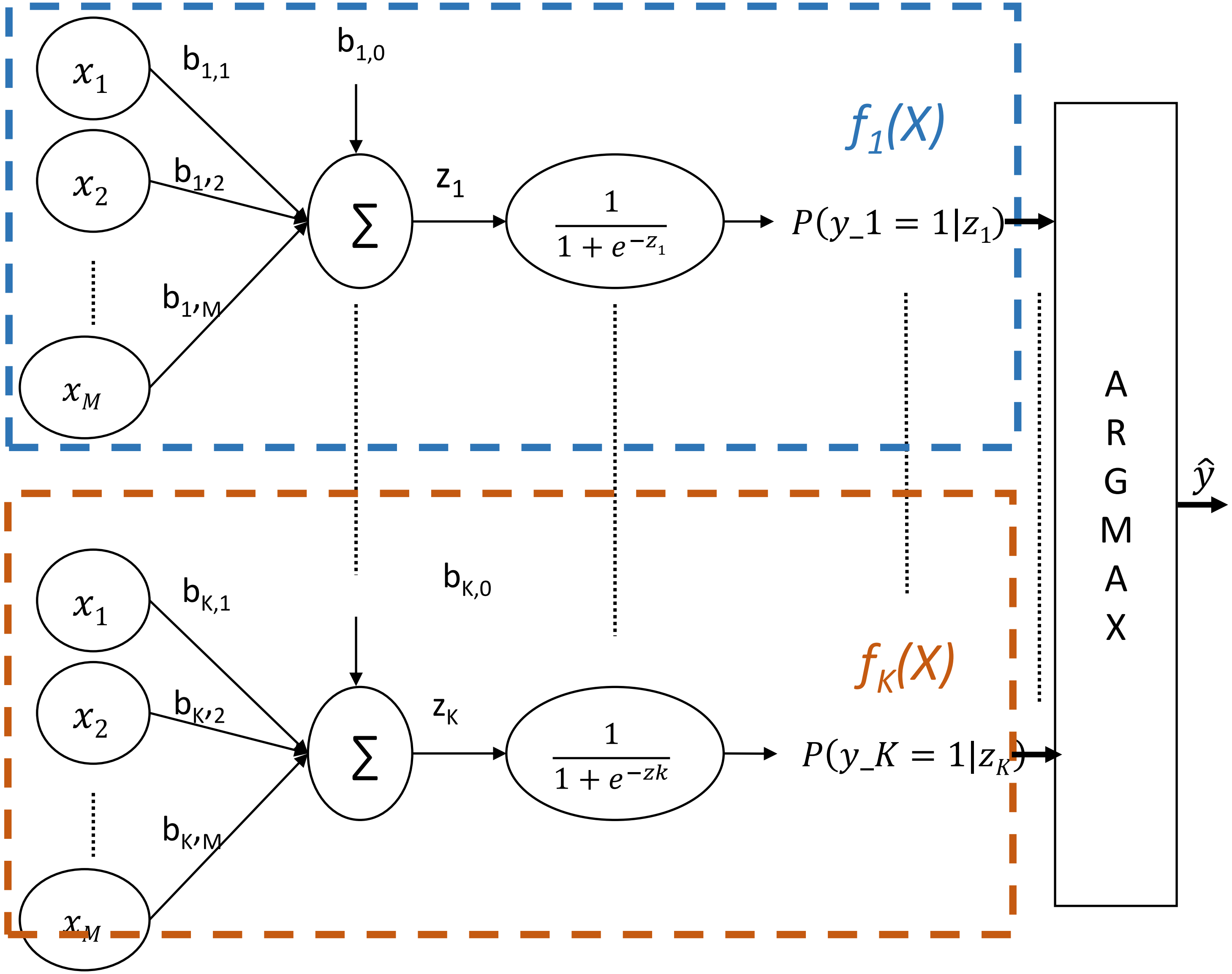
https://www.cs.rice.edu/~as143/COMP642_Spring22/Scribes/Lect5#:~:text=One%2DVs%2DAll%20Classification%20is,%2DAll%20multi%2Dclass%20classification.

Classification

Multiclass Classification – One-vs-Rest (One-vs-All)

- For $y \in \{1,2, \ldots, K\}$, there is a list of classifiers f_k for k $\in \{1,2, \ldots, K\}$
- The target variable y is encoded by one hot encoder into K columns (y_1, y_2, …y_K). One column for each class, which contains only zeros and ones
- Example: For classifier f_1*, the target variable is y_1. The probability of having 1, in this case, is the proba*bility, that the data point belongs to that class number 1.
- For each data point, apply all classifiers to X and predict the label k for which the corresponding classifier has the largest predicted probability

$$\hat{y} = \underset{k \in \{1,2, \ldots, K\}}{\operatorname{argmax}} f_k(X)$$

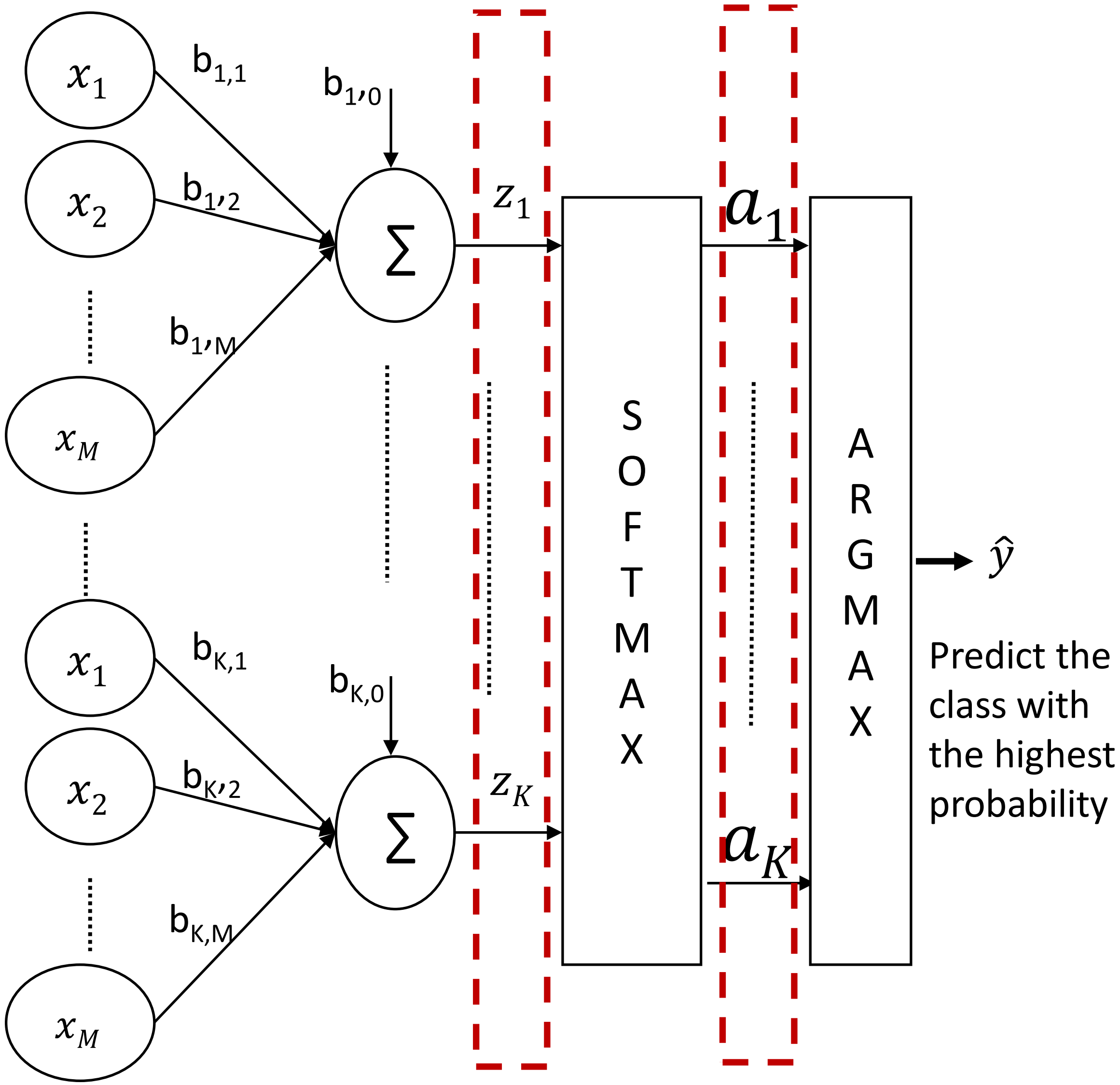


Classification

Multiclass Classification – Softmax Regression (Multinominal Logistic Regression)

+ In softmax regression the probability that a data point belongs to each class is calculated by softmax function instead of logistic function in logistic regression
+ The softmax function takes as input a vector Z of K real numbers ($Z=(z_1, z_2, \ldots, z_K)$)
+ $$softmax(z) = \left[\frac{e^{z_1}}{\sum_{j=1}^{K} e^{z_j}} \quad \frac{e^{z_2}}{\sum_{j=1}^{K} e^{z_j}} \cdots\cdots \frac{e^{z_K}}{\sum_{j=1}^{K} e^{z_j}} \right]$$
  $$= [\; a_1 \quad a_2 \quad \ldots\ldots \quad a_K \;]$$
+ Some of the elements of vector Z could be negative, or greater than one; and might not sum up to 1; but after applying softmax,
  + each component will be in the interval (0,1)

    $a_1, a_2, \ldots$ and $a_K \in [0,1]$
  + and they will sum up to 1, so that they can be interpreted as probabilities. $\sum_{j=1}^{K} a_j = 1$



Classification

Multiclass Classification – Softmax Regression (Multinominal Logistic Regression)

+ Softmax(z) can be represented also as:

  $$[P(y\_1 = 1|z_1)\ P(y\_2 = 1|z_2)\ \ldots\ldots.\ P(y\_k = 1|z_K)]$$

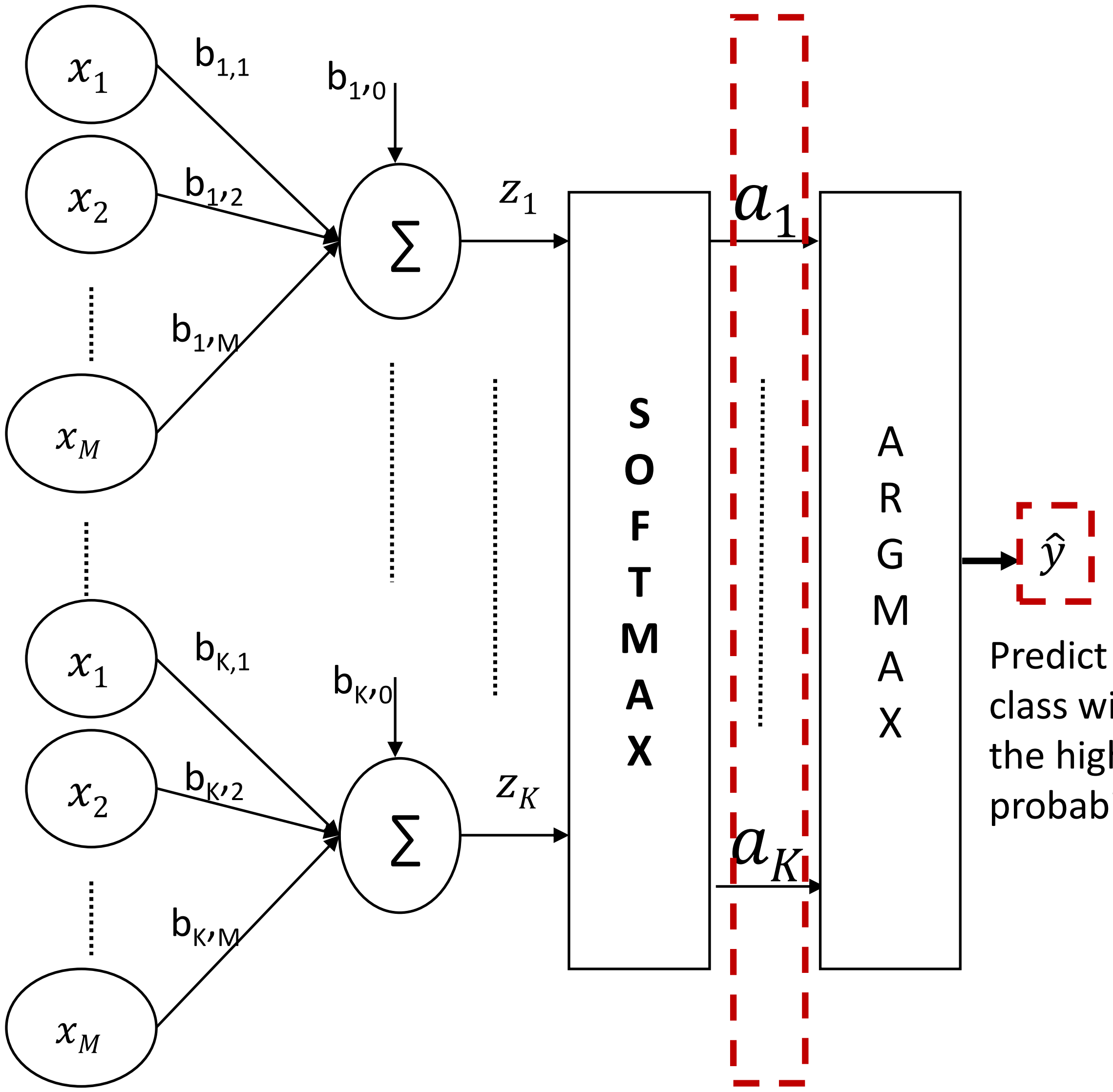
  Where (y_1, y_2, …y_K) are the encoded target variable y in one-hot encoder format

+ To make predictions, it is the argmax of the probabilities out of the softmax function

  $$\hat{y} = \underset{k \in \{1,2,\ldots.,K\}}{\text{argmax}}\ a_k$$

+ Because the softmax operation preserves the ordering among its arguments, we do not need to compute the softmax to determine which class has been assigned the highest probability. Thus,

$$\hat{y} = \underset{k \in \{1,2,\ldots.,K\}}{\text{argmax}}\ a_k = \underset{k \in \{1,2,\ldots.,K\}}{\text{argmax}}\ z_k$$



Predict the class with the highest probability

Classification

Multiclass Classification – Softmax Regression – Training

+ The element k of vector $Z^{(i)}$ of the training example number i can be represented as follows:

$$z_k^{(i)} = b_{k,0} + \sum_{m=1}^{M} b_{k,m} \cdot x_m^{(i)}$$

+ Sofmax($Z^{(i)}$) = $\left[a_1^{(i)} a_2^{(i)} \dots a_K^{(i)}\right]$

  $[P\left(y\_1^{(i)} = 1|z_1^{(i)}\right) P\left(y\_2^{(i)} = 1|z_2^{(i)}\right) \dots . P\left(y\_K^{(i)} = 1|z_K^{(i)}\right)]$

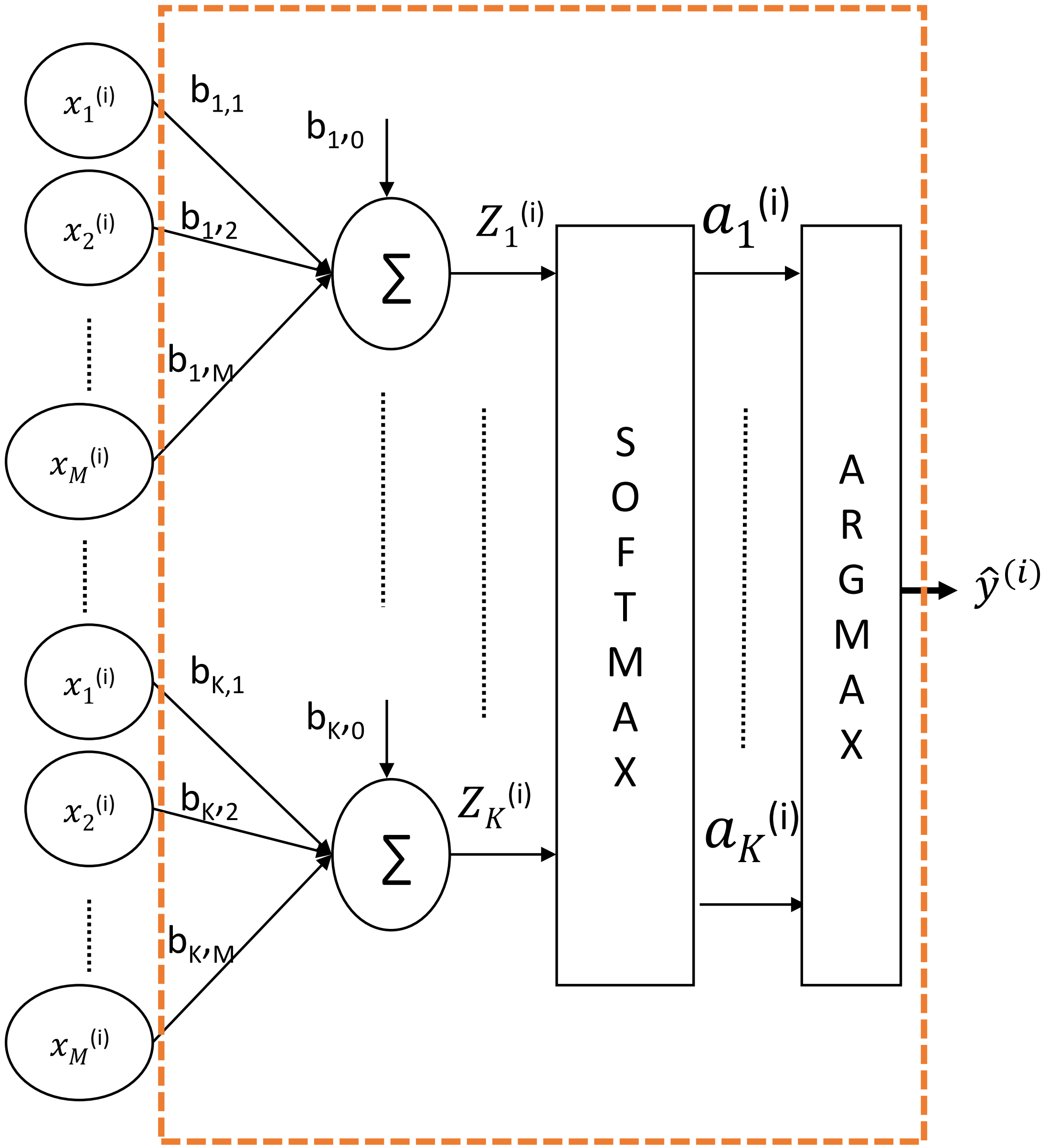
+ The probability that $y^{(i)}$ is assigned to a class (1, 2, .., or K) can be expressed as:

  $\prod_{k=1}^{K} P\left(y\_k^{(i)} = 1|z_k^{(i)}\right) = \prod_{k=1}^{K} (a_k^{(i)})^{y\_k^{(i)}}$

+ The Likelihood: Overall probability from the product of the probabilities of all the n trainings examples

  L = $\prod_{i=1}^{n} \prod_{k=1}^{K} (a_k^{(i)})^{y\_k^{(i)}}$

Training Example Number i



Classification

Multiclass Classification – Softmax Regression – Training

+ Simplifying the expression by taking logarithms

  $\mathrm{Ln(L)} = \sum_{i=1}^{n} \sum_{k=1}^{K} (y\_k^{(i)}) \ln(a_k^{(i)})$

+ Optimization by maximizing ln(L) or minimizing (- ln(L))

  Loss = $-\sum_{i=1}^{n} \sum_{k=1}^{K} (y\_k^{(i)}) \ln(a_k^{(i)})$

+ This loss function is called multi-category cross entropy

+ The model's parameter values $b_{k,m}$ can be estimated using a gradient descent method via partial derivatives and the multivariable chain rule, as an example for $b_{1,m}$:

$$\frac{\partial Loss}{\partial b_{1,m}} = \frac{\partial Loss}{\partial a_1} \cdot \frac{\partial a_1}{\partial z_1} \cdot \frac{\partial z_1}{\partial b_{1,m}} + \frac{\partial Loss}{\partial a_2} \cdot \frac{\partial a_2}{\partial z_1} \cdot \frac{\partial z_1}{\partial b_{1,m}} + \cdots + \frac{\partial Loss}{\partial a_K} \cdot \frac{\partial a_K}{\partial z_1} \cdot \frac{\partial z_1}{\partial b_{1,m}}$$

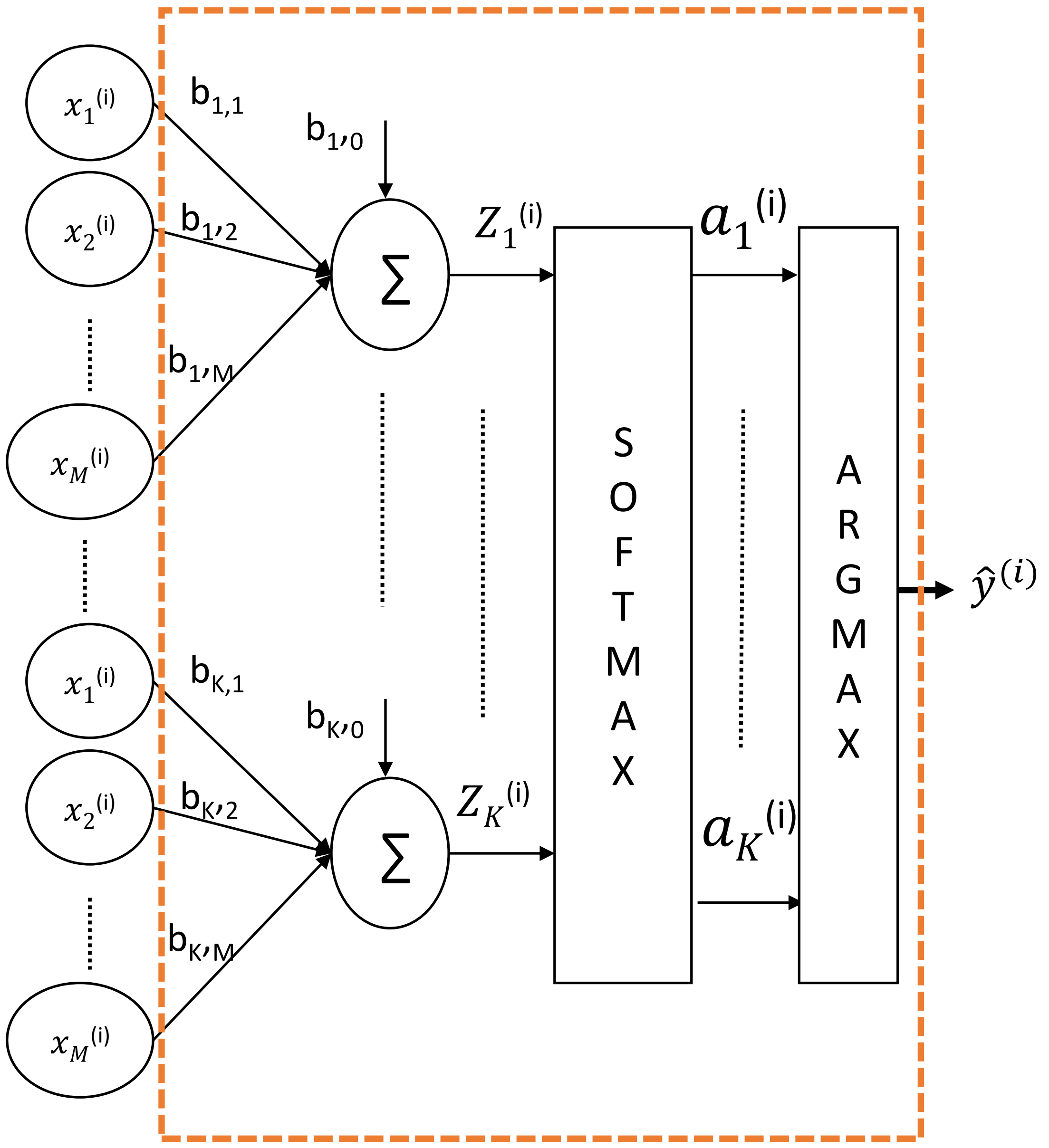
$$= -(y\_1 - a_1)\, x_m$$

The Error

Feature value

See step by step proof on ILIAS

Training Example Number i



Classification

Multiclass Classification – Softmax Regression – Training

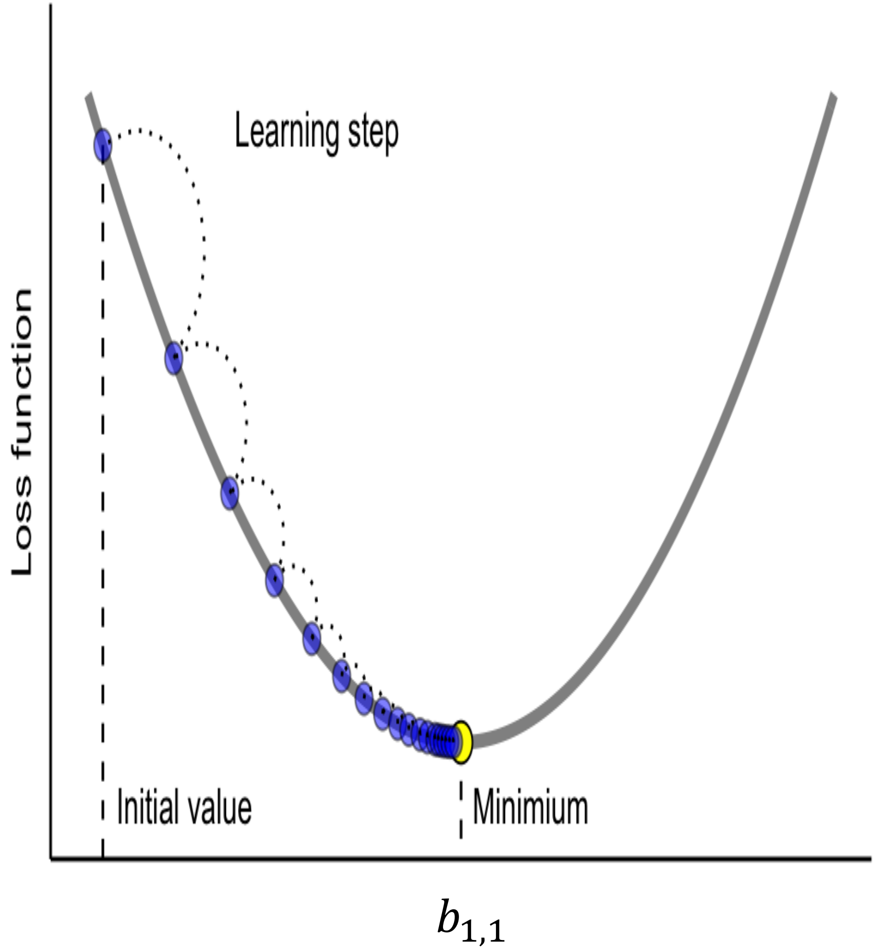
+ The steps involved in Gradient Descent:
  + Initialize values for the parameters $b_{k,m}^{[0]}$ to get started with the iteration process
  + Keep on iterating for d = 0, 1, 2,……. using the update rule of the Gradient Descent

$$b_{k,m}^{[d+1]} := b_{k,m}^{[d]} - \eta \cdot \frac{\partial Loss}{\partial b_{k,m}}$$

$$:= b_{k,m}^{[d]} - \eta . \sum_{i=1}^{n} (a_k^{(i)} - (y\_k^{(i)}) \cdot x_m$$

  η is called the learning rate or the learning step size

  + Termination criteria for a process can include:
    - Setting a specific number of iterations to be performed (number of epochs)
    - predefine improvement to be obtained in successive iterations
+ The learning rate and the number of epochs can be considered as hyperparameters of this method

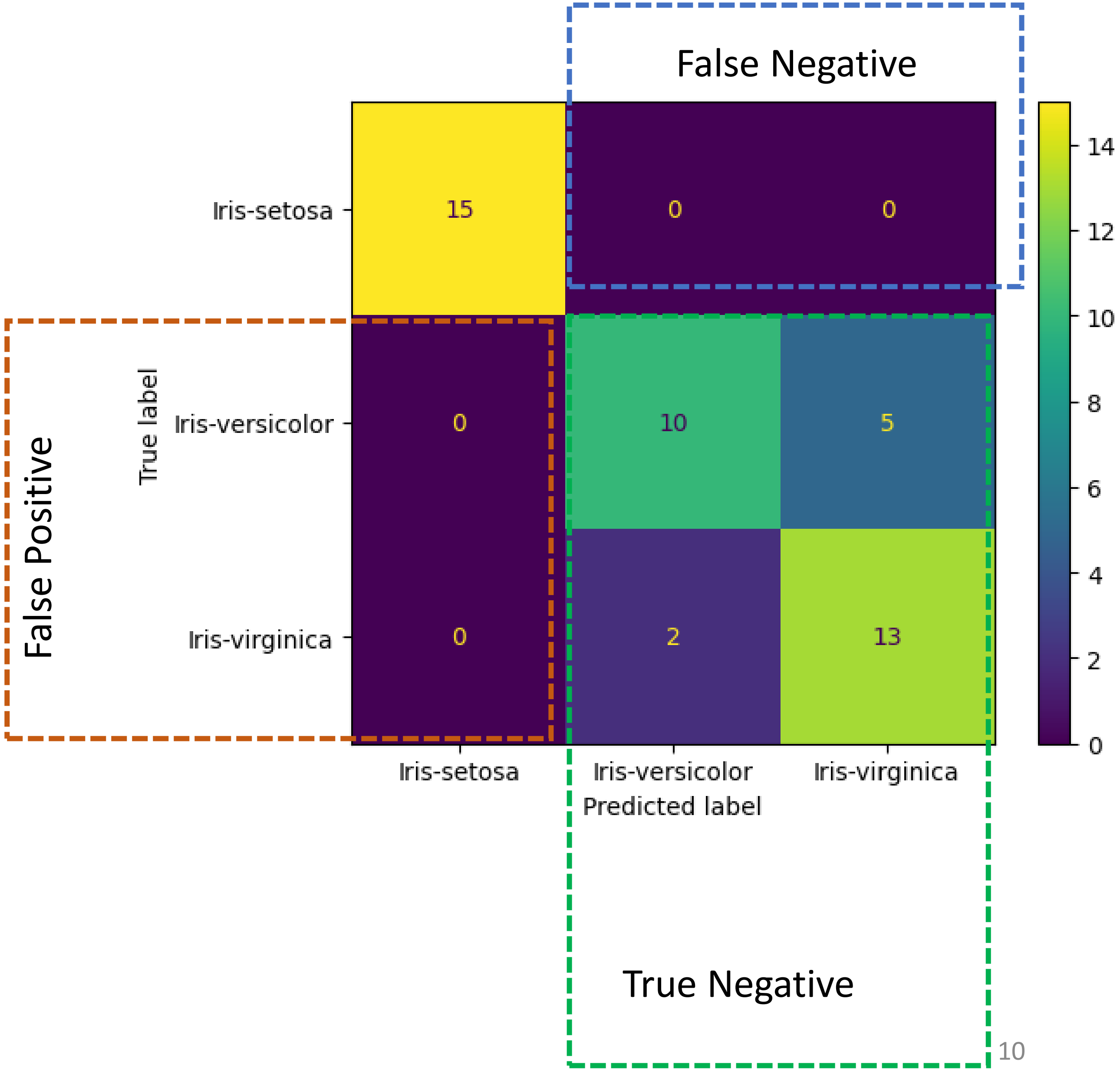


Classification

Multiclass Classification – Evaluation

+ The Confusion matrix for classification of IRIS dataset
+ For class Iris-Setosa:

+ True Positive: 15
+ False Negative (Type 2 Error): 0
+ False Positive (Type 1 Error): 0
+ True Negative: 30
+ Precision = TP/(TP+FP) = 1
+ Recall = TP/(TP+FN) = 1
+ F1 = 2*(1*1)/(1+1) =1



Classification

Multiclass Classification – Evaluation

+ Classification report is computed per class
+ Macro Average: It is referred to as the unweighted mean of the measure for each class.
    + Macro Precision = (1.00 + 0.83 + 0.72)/3 = 0.85

+ Unlike macro, it is the weighted mean of the measure. Weights are the total number of samples per class. In our example, we have 15 for every class
    + Weighted Precision = (15* 1.00 + 15* 0.83 + 15* 0.72)/45 = 0.85

	precision	recall	f1-score	support
Iris-setosa	1.00	1.00	1.00	15
Iris-versicolor	0.83	0.67	0.74	15
Iris-virginica	0.72	0.87	0.79	15
accuracy			0.84	45
macro avg	0.85	0.84	0.84	45
weighted avg	0.85	0.84	0.84	45

Classification

Multiclass Classification – Evaluation

+ Try to have different numbers for each class in support and compute again: Tipp: delete the argument (stratify=y) during data splitting
+ Macro Average: It is referred to as the unweighted mean of the measure for each class.
  + Macro Precision = (1.00 + 1.00 + 0.68)/3 = 0.89
+ Unlike macro, it is the weighted mean of the measure. Weights are the total number of samples per class. In this example, we have 19 for setose, 13 versicolor and 13 virginica
  + Weighted Precision = (19* 1.00 + 13* 1.00 + 13* 0.68)/45 = 0.91

	precision	recall	f1-score	support
Iris-setosa	1.00	1.00	1.00	19
Iris-versicolor	1.00	0.54	0.70	13
Iris-virginica	0.68	1.00	0.81	13
accuracy			0.87	45
macro avg	0.89	0.85	0.84	45
weighted avg	0.91	0.87	0.86	45

Hochschule Karlsruhe
University of
Applied Sciences

Fakultät für
Elektro- und
Informationstechnik

www.h-ka.de