



UNIVERSITÄT
ZU KÖLN

Machine Learning: Logistic Regression

VL Sprachliche Informationsverarbeitung

Nils Reiter

`nils.reiter@uni-koeln.de`

December 7, 2022

Winter term 2023/24

Recap: Machine Learning so Far

- ▶ Naive Bayes
 - ▶ “Classical machine learning”
 - ▶ Binary classification method
 - ▶ Manually defined features (e.g., some tokens appearing in an e-mail)
- ▶ Evaluation
 - ▶ Accuracy, precision, recall, F-score
 - ▶ Baseline

Section 1

Linear/Logistic Regression

Regression

- ▶ Regression
 - ▶ Prediction of numeric values (e.g., future COVID-19 cases; number of nouns in a text, ...)
 - ▶ Based on some input features (e.g., “R-Wert”, number of past cases, ...)

Regression

- ▶ Regression
 - ▶ Prediction of numeric values (e.g., future COVID-19 cases; number of nouns in a text, ...)
 - ▶ Based on some input features (e.g., “R-Wert”, number of past cases, ...)
- ▶ Linear
 - ▶ The relation between input features and output values is linear
 - ▶ Math: $y = a_1x_1 + a_2x_2 + \dots + a_nx_n + b$

Regression

- ▶ Regression
 - ▶ Prediction of numeric values (e.g., future COVID-19 cases; number of nouns in a text, ...)
 - ▶ Based on some input features (e.g., “R-Wert”, number of past cases, ...)
- ▶ Linear
 - ▶ The relation between input features and output values is linear
 - ▶ Math: $y = a_1x_1 + a_2x_2 + \dots + a_nx_n + b$
- ▶ Logistic
 - ▶ Relation between input and output follows a logistic equation σ :
 - ▶ $0 \leq \sigma(x) \leq 1$, for all values of x
 - ▶ They can be interpreted as probabilities

Linear Regression

Example

- ▶ Input: Number of words in a (narrative, prose) text
- ▶ Output: Number of literary characters in the text
(in the sense of “Figur”, not in the sense of “Zeichen”)

Linear Regression

Example

- ▶ Input: Number of words in a (narrative, prose) text
- ▶ Output: Number of literary characters in the text
(in the sense of “Figur”, not in the sense of “Zeichen”)
- ▶ Linear equation (with one input variable): $y = ax + b$
 - ▶ With x being the number of tokens and y the number of characters

Linear Regression

The data set

x (# pages)	y (# characters)
10	3
105	5
150	8
210	12
250	7
295	13

Linear Regression

The data set

x (# pages)	y (# characters)
10	3
105	5
150	8
210	12
250	7
295	13

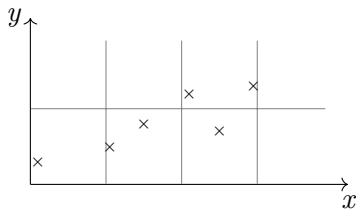


Figure: Data set, each \times represents a text (x : text length, y : num. of characters)

Linear Regression

The data set

x (# pages)	y (# characters)
10	3
105	5
150	8
210	12
250	7
295	13

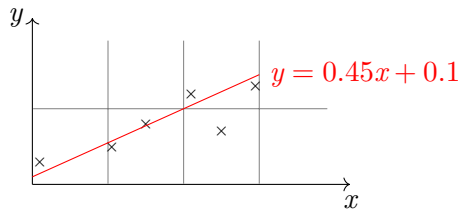
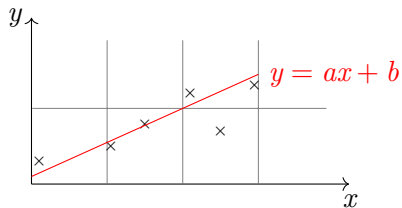


Figure: Data set, each \times represents a text (x : text length, y : num. of characters)

Linear Regression

The Task



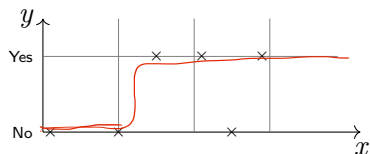
The Model

- ▶ Linear regression with one variable (= univariate linear regression)
- ▶ Prediction (hypothesis function): $y = h_{a,b}(x) = ax + b$
- ▶ How to set parameters a and b ? → training algorithm

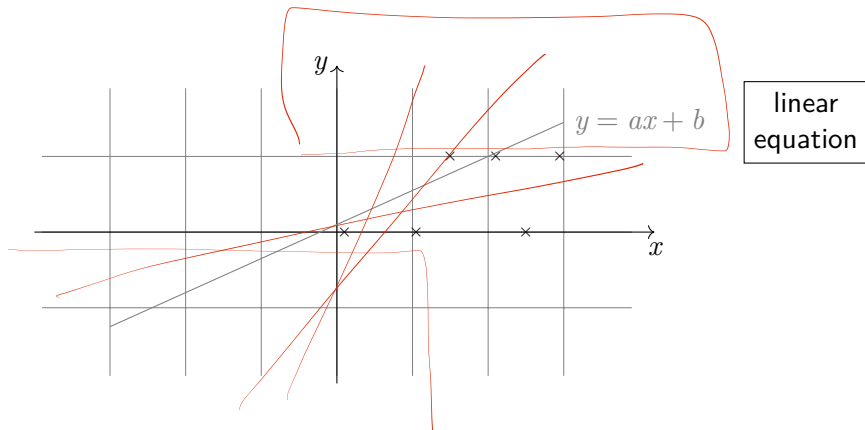
From Regression to Classification

- ▶ Example task: Given the number of words in an e-mail, is it spam?

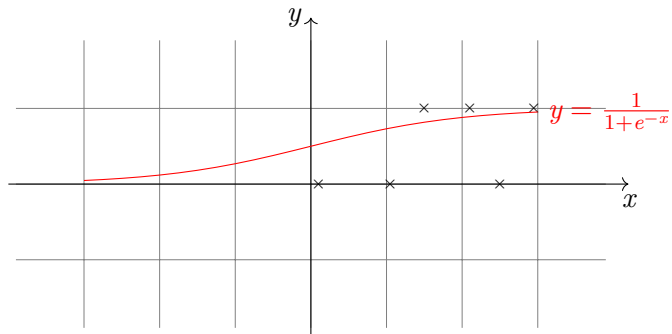
x (# words)	y (# spam)
10	No
100	No
150	Yes
210	Yes
250	No
290	Yes



Fitting an Equation

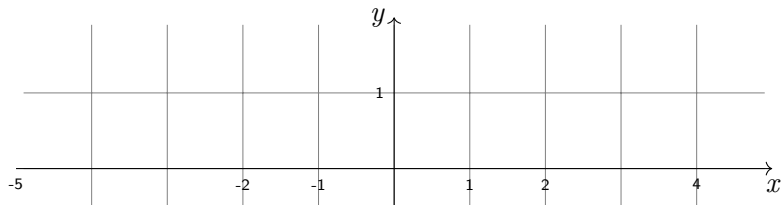


Fitting an Equation



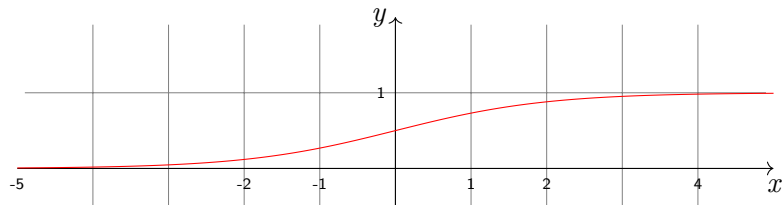
logistic
function

The Logistic Function



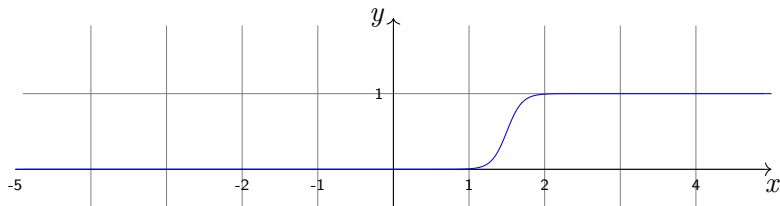
$$y = \frac{1}{1 + e^{-(ax+b)}} \quad (\text{general form})$$

The Logistic Function



$$y = \frac{1}{1 + e^{-(ax+b)}} \quad (\text{general form})$$
$$y = \frac{1}{1 + e^{-(1*x+0)}}$$

The Logistic Function

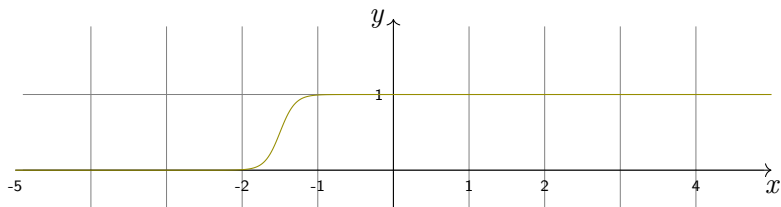


$$y = \frac{1}{1+e^{-(ax+b)}} \quad (\text{general form})$$

$$y = \frac{1}{1+e^{-(1*x+0)}}$$

$$y = \frac{1}{1+e^{-(10*x-15)}}$$

The Logistic Function



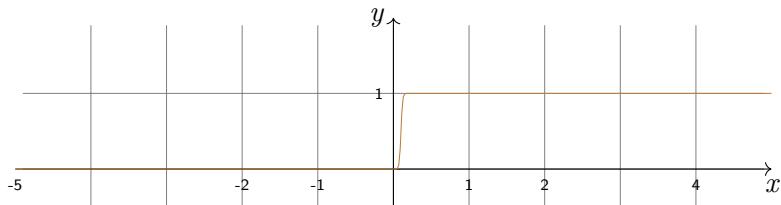
$$y = \frac{1}{1+e^{-(ax+b)}} \quad (\text{general form})$$

$$y = \frac{1}{1+e^{-(1*x+0)}}$$

$$y = \frac{1}{1+e^{-(10*x-15)}}$$

$$y = \frac{1}{1+e^{-(10*x+15)}}$$

The Logistic Function



$$y = \frac{1}{1+e^{-(ax+b)}} \quad (\text{general form})$$

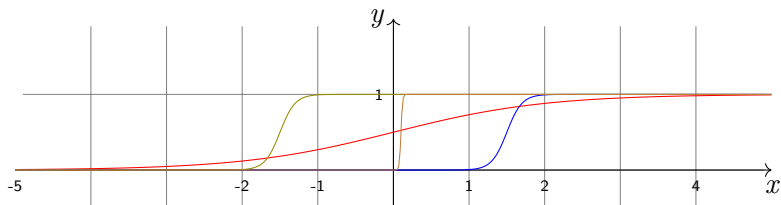
$$y = \frac{1}{1+e^{-(1*x+0)}}$$

$$y = \frac{1}{1+e^{-(10*x-15)}}$$

$$y = \frac{1}{1+e^{-(10*x+15)}}$$

$$y = \frac{1}{1+e^{-(100*x-10)}}$$

The Logistic Function



$$y = \frac{1}{1+e^{-(ax+b)}} \quad (\text{general form})$$

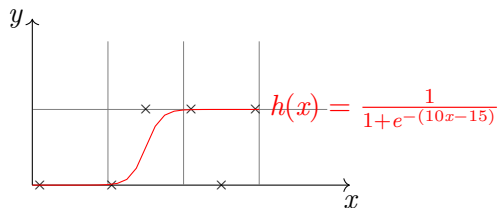
$$y = \frac{1}{1+e^{-(1*x+0)}}$$

$$y = \frac{1}{1+e^{-(10*x-15)}}$$

$$y = \frac{1}{1+e^{-(10*x+15)}}$$

$$y = \frac{1}{1+e^{-(100*x-10)}}$$

Parameter Fitting



- ▶ Linear equations can be wrapped in a logistic one

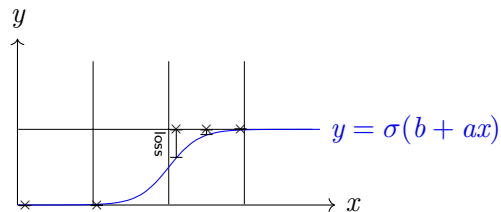
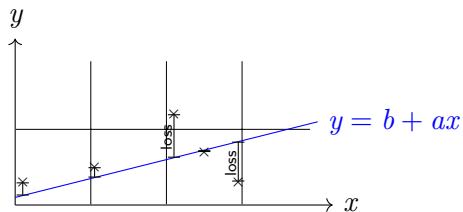
- ▶ $\sigma(x) = \frac{1}{1+e^{-x}} \rightarrow y = \sigma(ax + b)$

- ▶ Same parameters to be tuned (a and b)

- ▶ $e = \sum_{n=0}^{\infty} \frac{1}{n!} = 2.71828$ (Euler's number)

Summary: Linear/Logistic Regression

(With a single variable)



- ▶ Logistic/linear regression is half of the math of deep learning
- ▶ Remaining question: How to learn parameters a and b ?

**SPOILER
ALERT!**

Generalization to More Parameters

- ▶ Example: Classify Spam/Not Spam based on number of tokens, but: number of words from keyword list (e.g., “profit”)
- ▶ Features: x_1, x_2, x_3, \dots
- ▶ Logistic regression model: $y = \sigma(w_0 + w_1x_1 + w_2x_2 + w_3x_3)$
 - ▶ I.e., one w parameter for each feature, and an additional constant w_0
- ▶ Parameter vector: $\vec{w} = \langle w_0, w_1, w_2, \dots \rangle$
- ▶ Input vector: $\vec{x} = \langle x_1, x_2, x_3, \dots \rangle$

Section 2

Practice

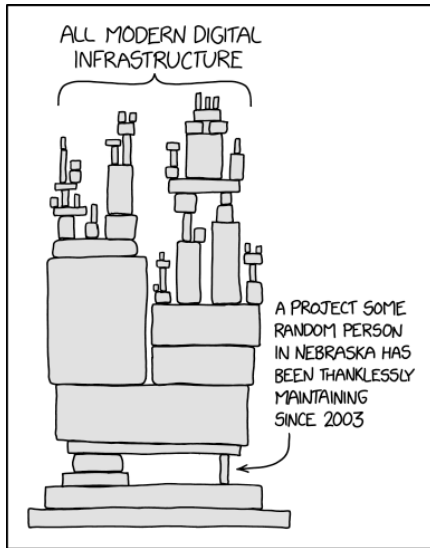


Figure: XKCD 2347

Doing Logistic Regression in Practice

- ▶ Many implementations in for many programming languages
- ▶ Python `scikit-learn`
- ▶ General structure (convention)
 - ▶ Data loading
 - ▶ Typical variable names: `x_train`, `y_train`, `x_test`, `y_test`
 - ▶ Preprocessing
 - ▶ Model initialisation (or loading if its pre-trained)
 - ▶ Model training (function often called `fit()`)
 - ▶ Model evaluation

demo

Hausaufgabe 4 (bis 21.12.)



Trainieren Sie ein logistisches Regressionsmodell, um handgeschriebene Ziffern zu erkennen. Die Ziffern wurden handgeschrieben, schwarz/weiß eingescannt und die Bilder dann als 28x28-Matrizen mit Graustufeninformationen bereitgestellt. Es handelt sich nur um Nullen und Einsen, und ist damit eine binäre Klassifikationsaufgabe. Sie finden die Trainings- und Testdaten hier, und hier ein Python-Skript, mit einer Funktion zum Einlesen der Daten. Verwenden Sie die Bibliothek scikit-learn für das eigentliche Training (und schauen Sie sich ruhig ein bisschen um, was die Bibliothek sonst so bereithält).

Section 3

Gradient Descent

Learning Regression Models

- ▶ How to select the parameters w_0, w_1 such that the hypothesis function describes the data points as best as possible?
- ▶ Learning algorithm *Gradient Descent*

Learning Regression Models

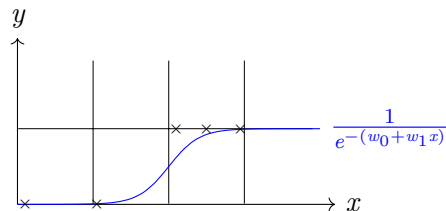
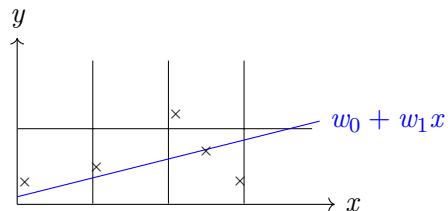
- ▶ How to select the parameters w_0, w_1 such that the hypothesis function describes the data points as best as possible?
- ▶ Learning algorithm *Gradient Descent*



Gradient descent is half of the algorithms of deep learning

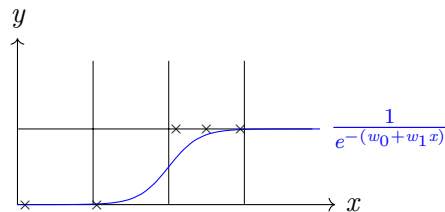
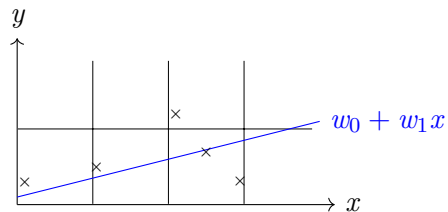
Loss: Intuition

The *loss* measures the 'wrongness' of values for w_0 and w_1



Loss: Intuition

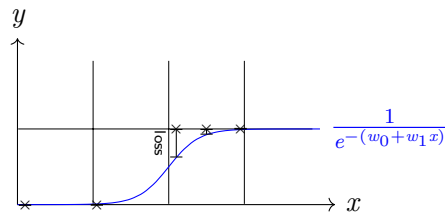
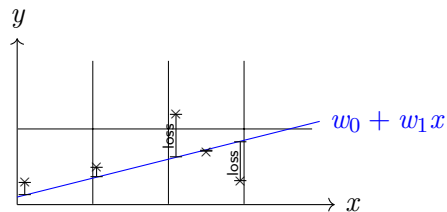
The *loss* measures the 'wrongness' of values for w_0 and w_1



- ▶ How big is the gap between a hypothesis (= the) and the data?
- ▶ Is $\vec{w} = \langle 0.3, 0.5 \rangle$ or $\vec{w} = \langle 0.4, 0.4 \rangle$ better?

Loss: Intuition

The *loss* measures the 'wrongness' of values for w_0 and w_1

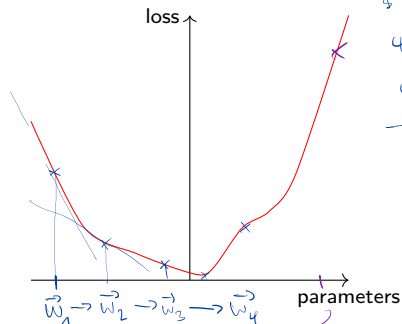


- ▶ How big is the gap between a hypothesis (= the) and the data?
- ▶ Is $\vec{w} = \langle 0.3, 0.5 \rangle$ or $\vec{w} = \langle 0.4, 0.4 \rangle$ better?

Gradient Descent: Intuition

- ▶ Loss should be as small as possible
- ▶ Total loss can be calculated for given parameters $\vec{w} = (w_0, w_1)$ (and with respect to a data set!)
- ▶ Idea:
 - ▶ Make many iterations
 - ▶ In each iteration, change \vec{w} towards to make the loss smaller
 - ▶ We use the derivative of the loss function to know how \vec{w} needs to be changed

Loss function: Intuition



1. Iteration $\vec{w} = \langle 0.8, -8.3 \rangle$

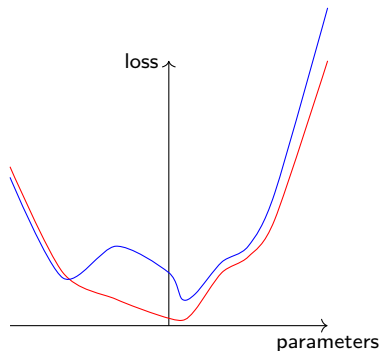
2. Iteration $\vec{w} = \langle 0.5, -3.8 \rangle$

3. \vdots

4. \vdots

5. Iteration $\vec{w} = \langle 0.1, -2.7 \rangle$

Loss function: Intuition



Function should be **convex**!

If not, we might get stuck in local minimum

Loss Function with More Parameters

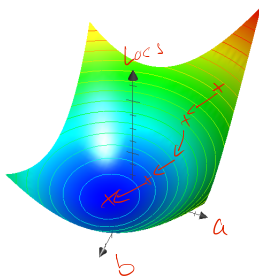


Figure: The loss function in our setting visualised

- ▶ Searching for the a, b settings with minimal loss
- ▶ = Searching for the minimum!

Hypothesis vs. Loss Function

Hypothesis function h

$$\sigma(ax + b)$$

- ▶ Calculates outcome for a single instance based on
 - ▶ Feature values \vec{x}
 - ▶ Parameter values \vec{w}

Loss function J

- ▶ Calculates 'wrongness' of h based on
 - ▶ Parameter values \vec{w}
 - ▶ A data set consisting of many instances with
 - ▶ Feature values \vec{x}
 - ▶ Correct outcomes Y

Loss Function

Definition

Loss function depends on hypothesis function

Linear hypothesis function

- ▶ $h(x) = w_0 + w_1x_1$
- ▶ Loss: Mean squared error

Loss Function

Definition

Loss function depends on hypothesis function

Linear hypothesis function

- ▶ $h(x) = w_0 + w_1 x_1$
- ▶ Loss: Mean squared error

Logistic hypothesis function

- ▶ $h(x) = \sigma(w_0 + w_1 x_1) = \frac{1}{e^{-(w_0 + w_1 x_1)} + 1}$
- ▶ Loss: (Binary) cross-entropy loss

Loss Function

Definition for Linear Regression

- ▶ The loss function is a function on parameter values a and b
(for a given hypothesis function and data set)
 - ▶ Hypothesis function: $h_{\vec{w}} = w_1 x + w_0$

$\vec{w} = (w_0, w_1)$: parameters $h_{\vec{w}}$: hypothesis function m : number of items

$$J(\vec{w}) =$$

Loss Function

Definition for Linear Regression

- ▶ The loss function is a function on parameter values a and b (for a given hypothesis function and data set)
 - ▶ Hypothesis function: $h_{\vec{w}} = w_1 x + w_0$

$\vec{w} = (w_0, w_1)$: parameters $h_{\vec{w}}$: hypothesis function m : number of items

$$J(\vec{w}) = \quad h_{\vec{w}}(x_i) - y_i$$

- ▶ Calculate the loss for item i

Loss Function

Definition for Linear Regression

- ▶ The loss function is a function on parameter values a and b (for a given hypothesis function and data set)
 - ▶ Hypothesis function: $h_{\vec{w}} = w_1 x + w_0$

$\vec{w} = (w_0, w_1)$: parameters $h_{\vec{w}}$: hypothesis function m : number of items

$$J(\vec{w}) = \sum_{i=1}^m (h_{\vec{w}}(x_i) - y_i)^2$$

- ▶ Calculate the loss for item i
- ▶ Square the error

Loss Function

Definition for Linear Regression

- ▶ The loss function is a function on parameter values a and b (for a given hypothesis function and data set)
 - ▶ Hypothesis function: $h_{\vec{w}} = w_1 x + w_0$

$\vec{w} = (w_0, w_1)$: parameters $h_{\vec{w}}$: hypothesis function m : number of items

$$J(\vec{w}) = \sum_{i=1}^m (h_{\vec{w}}(x_i) - y_i)^2$$

- ▶ Calculate the loss for item i
- ▶ Square the error
- ▶ Sum them up

Loss Function

Definition for Linear Regression

- ▶ The loss function is a function on parameter values a and b (for a given hypothesis function and data set)
 - ▶ Hypothesis function: $h_{\vec{w}} = w_1 x + w_0$

$\vec{w} = (w_0, w_1)$: parameters $h_{\vec{w}}$: hypothesis function m : number of items

$$J(\vec{w}) = \frac{1}{m} \sum_{i=1}^m (h_{\vec{w}}(x_i) - y_i)^2$$

- ▶ Calculate the loss for item i
- ▶ Square the error
- ▶ Sum them up
- ▶ Divide by the number of items
 - ▶ Known as: *Mean squared error*

Loss Function

Definition for Linear Regression

- ▶ The loss function is a function on parameter values a and b
(for a given hypothesis function and data set)
 - ▶ Hypothesis function: $h_{\vec{w}} = w_1 x + w_0$

$\vec{w} = (w_0, w_1)$: parameters $h_{\vec{w}}$: hypothesis function m : number of items

$$J(\vec{w}) = \frac{1}{2} \frac{1}{m} \sum_{i=1}^m (h_{\vec{w}}(x_i) - y_i)^2$$

- ▶ Calculate the loss for item i
- ▶ Square the error
- ▶ Sum them up
- ▶ Divide by the number of items
 - ▶ Known as: *Mean squared error*
- ▶ Divide by two
 - ▶ out of convenience, because derivation

Loss function

Definition for Logistic Regression

- ▶ Two cases: $y_i = 0$ or $y_i = 1 - y_i$: real outcome for instance i
 - ▶ I.e., for each instance, one of the summands is 0

Loss function

Definition for Logistic Regression

- ▶ Two cases: $y_i = 0$ or $y_i = 1 - y_i$: real outcome for instance i
 - ▶ I.e., for each instance, one of the summands is 0

$$J(\vec{w}) = \sum_i \left(y_i \log(h_{\vec{w}}(x_i)) + (1 - y_i) \log(1 - h_{\vec{w}}(x_i)) \right)$$

Loss function

Definition for Logistic Regression

- ▶ Two cases: $y_i = 0$ or $y_i = 1 - y_i$: real outcome for instance i
 - ▶ I.e., for each instance, one of the summands is 0

$$J(\vec{w}) = \log h_{\vec{w}}(x_i) + \log(1 - h_{\vec{w}}(x_i))$$

Loss function

Definition for Logistic Regression

- ▶ Two cases: $y_i = 0$ or $y_i = 1 - y_i$: real outcome for instance i
 - ▶ I.e., for each instance, one of the summands is 0

$$J(\vec{w}) = y_i \log h_{\vec{w}}(x_i) + (1 - y_i) \log (1 - h_{\vec{w}}(x_i))$$

Loss function

Definition for Logistic Regression

- ▶ Two cases: $y_i = 0$ or $y_i = 1 - y_i$: real outcome for instance i
 - ▶ I.e., for each instance, one of the summands is 0

$$J(\vec{w}) = -\frac{1}{m} \sum_{i=0}^m y_i \log h_{\vec{w}}(x_i) + (1 - y_i) \log (1 - h_{\vec{w}}(x_i))$$

$$= -\frac{1}{m} \sum_{i=0}^m \begin{cases} \log h_w(x_i) & \text{if } y_i = 1 \\ \log 1 - h_w(x_i) & \text{if } y_i = 0 \end{cases}$$

Loss function

Definition for Logistic Regression

- ▶ Two cases: $y_i = 0$ or $y_i = 1 - y_i$: real outcome for instance i
 - ▶ I.e., for each instance, one of the summands is 0

$$J(\vec{w}) = -\frac{1}{m} \sum_{i=0}^m \underbrace{y_i \log h_{\vec{w}}(x_i)}_{0 \text{ iff } y_i=0} + \underbrace{(1 - y_i) \log(1 - h_{\vec{w}}(x_i))}_{0 \text{ iff } y_i=1}$$

Loss function

Definition for Logistic Regression

- ▶ Two cases: $y_i = 0$ or $y_i = 1 - y_i$: real outcome for instance i
 - ▶ I.e., for each instance, one of the summands is 0

y_i	$h_{\vec{w}}(x_i) + \epsilon$	$y_i \log h_{\vec{w}}(x_i) + (1 - y_i) \log(1 - h_{\vec{w}}(x_i))$
0	1.0000001	-23.2535
0	0	0

Loss function

Definition for Logistic Regression

- ▶ Two cases: $y_i = 0$ or $y_i = 1 - y_i$: real outcome for instance i
 - ▶ I.e., for each instance, one of the summands is 0

y_i	$h_{\vec{w}}(x_i) + \epsilon$	$y_i \log h_{\vec{w}}(x_i) + (1 - y_i) \log(1 - h_{\vec{w}}(x_i))$
0	1.0000001	-23.2535
0	0	0
1	1	0
1	0.0000001	-23.2535

Loss function

Definition for Logistic Regression

- ▶ Two cases: $y_i = 0$ or $y_i = 1 - y_i$: real outcome for instance i
 - ▶ I.e., for each instance, one of the summands is 0

y_i	$h_{\vec{w}}(x_i) + \epsilon$	$y_i \log h_{\vec{w}}(x_i) + (1 - y_i) \log(1 - h_{\vec{w}}(x_i))$
0	1.0000001	-23.2535
0	0	0
1	1	0
1	0.0000001	-23.2535
1	0.8	-0.3219281
1	0.2	-2.321928

Caveat: $\log 0$ is undefined – add $\epsilon = 0.0000001$ if needed

Misunderstandings about Loss

- ▶ Loss values are not an evaluation measure
- ▶ It is meaningless to report loss values in the same way as accuracy or precision

Misunderstandings about Loss

- ▶ Loss values are not an evaluation measure
- ▶ It is meaningless to report loss values in the same way as accuracy or precision
- ▶ Because it's not scaled
- ▶ When used to compare, scale not important

Misunderstandings about Loss

- ▶ Loss values are not an evaluation measure
- ▶ It is meaningless to report loss values in the same way as accuracy or precision
- ▶ Because it's not scaled
- ▶ When used to compare, scale not important

Example

x	y
10	3
105	5
150	8
210	12
250	7
295	13

Table: Lower Loss Value

x	y
10	30
105	50
150	80
210	120
250	70
295	130

Table: Higher Loss Value

Summary

Regression

- ▶ Fitting parameters to a data distribution
 - ▶ Linear R: Numeric prediction algorithm
 - ▶ Prediction model: $h_{\vec{w}}(x) = w_0 + w_1 x$
 - ▶ Logistic R: Classification algorithm (because we interpret results as probabilities)
 - ▶ Prediction model: $h_{\vec{w}}(x) = \frac{1}{e^{-(w_0 + w_1 x)}}$
- ▶ Learning algorithm: Gradient descent

Gradient Descent

- ▶ Initialise \vec{w} with random values (e.g., 0)
- ▶ Repeat:
 - ▶ Find the direction to the minimum by taking the derivative
 - ▶ Change \vec{w} accordingly, using a learning rate η
 - ▶ Stop when \vec{w} don't change anymore

References I