EPFL

Lecture 03
30.09.2024

Regularized regression

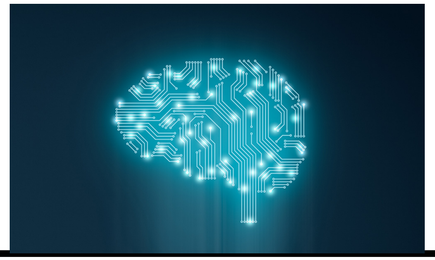
Classification through logistic regression

Outline

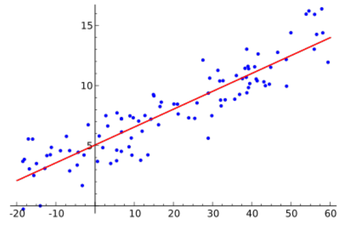
- Overfitting/underfitting and regularization
- Logistic regression

- Announcements:
  - Check moodle for past year's exams and quizzes

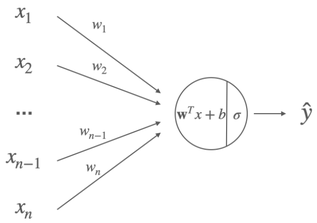
Introduction



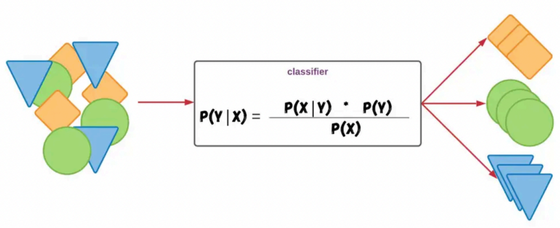
Linear regression



Logistic regression



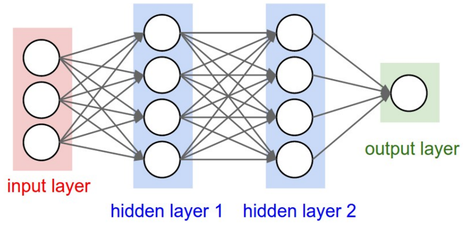
Naive Bayes



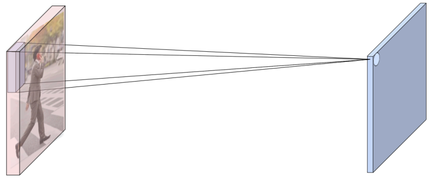
KNN

AI ethics

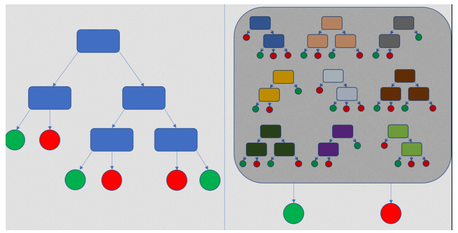
Neural networks



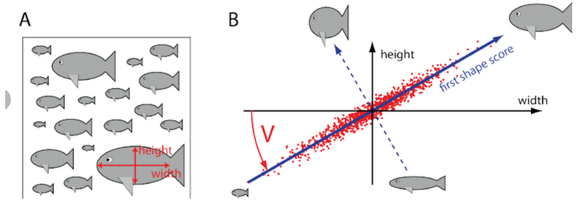
Convolutional neural networks



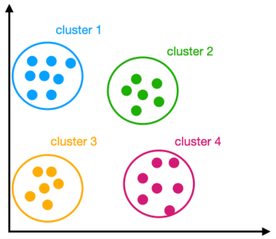
Decision-trees



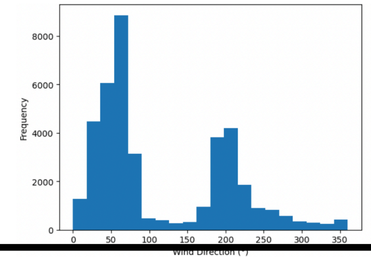
Dimensionality reduction



Clustering



Reinforcement learning



Review of linear regression

- Linear model: $\{x^i, y^i\}_{i=1}^N$, $x^i \in \mathbb{R}^d$, $y^i \in \mathbb{R}^m$

$$y = b + w_1 x_1 + \cdots + w_d x_d \qquad w = [b, w_1, \cdots, w_d] \in \mathbb{R}^{d+1}$$

$$\phi : \mathbb{R}^d \longrightarrow \mathbb{R}^p$$

$$y = b + w_1 \phi_1(x) + \cdots + w_p \phi_p(x)$$

- Mean-Square-Error (MSE) loss

$$\hat{y}^i = b + w_1 x_1^i + \cdots + w_d x_d^i$$

$$J(w) = \frac{1}{N} \sum_{i=1}^{N} (\hat{y}^i - y^i)^2$$

Underfitting and overfitting

Underfitting & Overfitting

Goal of supervised ML models: generalise well on new data (based on the patterns learned from known data).

Ways ML can fail

- Underfitting : model too simple, not enough data
- Overfitting

Underfitting & Overfitting

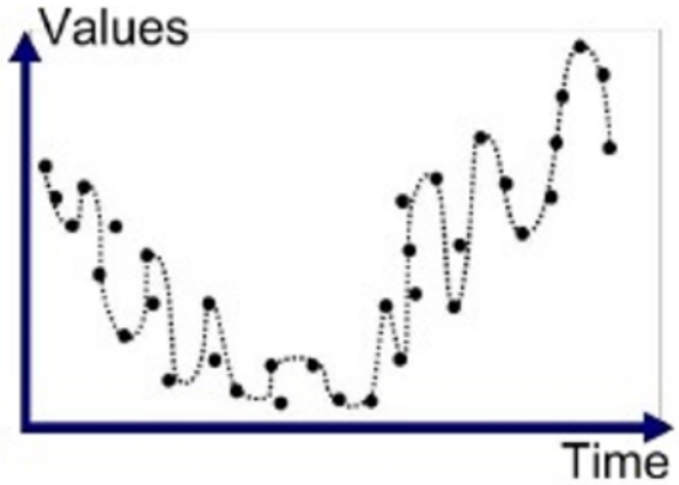
Overfitting

Q: What if the ML model is too complex?

Overfitting model:

- Fits well on training data
- Doesn't generalise well to unknown data.

Reason: the model is too complex → it fits the noises and errors.



The model passes by every data point but doesn't capture the U shape of the data set.

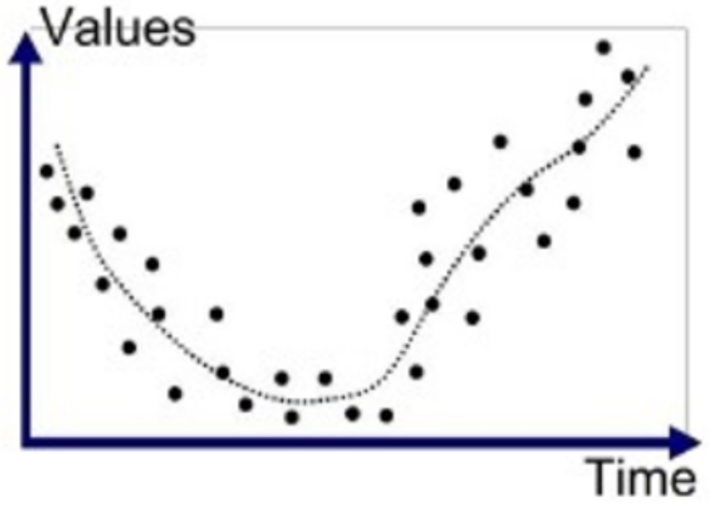
Underfitting & Overfitting

Overfitting - Solutions

Solutions:

- Simpler model → fit the data and not the noises and errors.
- More training data (→ less effect of noise)
- Add a regularisation term (common solution)

Goal: Find the equilibrium between fitting the training data and keeping the model simple enough to ensure it will generalise well on new data.



Sensitivity and Regularization

- Sensitivity of a predictor: similar data should lead to similar outcome
  - Less sensitive models tend to not overfit $x, x' \in \mathbb{R}^d$

similar data x, x' $\|x - x'\|_2^2 = (x_1 - x_1')^2 + (x_2 - x_2')^2 + \cdots + (x_d - x_d')^2$

$\|x - x'\|_2$ small x is similar to x'.

we would like $f_w : \mathbb{R}^d \longrightarrow \mathbb{R}$ to give similar

labels for x, x' :

$|f_w(x) - f_w(x')|$ be small.

Sensitivity of linear models

Motivation for regularization through Lipschitz constant of the predictor

Consider $f_w(x) = b + w_1 x_1 + \cdots + w_d x_d$

$$|f_w(x) - f_w(x')| = |w_1(x_1 - x_1') + w_2(x_2 - x_2') + \cdots + w_d(x_d - x_d')|$$

$$= \left| \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix}^T \begin{bmatrix} x_1 - x_1' \\ x_2 - x_2' \\ \vdots \\ x_d - x_d' \end{bmatrix} \right| \leq \left\| \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix} \right\|_2 \| x - x' \|_2$$

Cauchy-Schwarz inequality

$\Rightarrow$ if $\|w\|_2$ small then $f_w(x)$ will be close to $f_w(x')$ for x close to x'.

Exercise - optimize a regularized loss function

- Regularised loss function $J(w) = \frac{1}{N}\sum_{i=1}^{N}(w^T x^i - y^i)^2 + \lambda(w_1^2 + w_2^2 + \ldots + w_d^2)$

$\lambda \in \mathbb{R}_+$

- Find the gradient of the regularized loss function with respect to the parameters

parameters $w = \begin{bmatrix} b \\ w_1 \\ \vdots \\ w_d \end{bmatrix} \in \mathbb{R}^{d+1}$

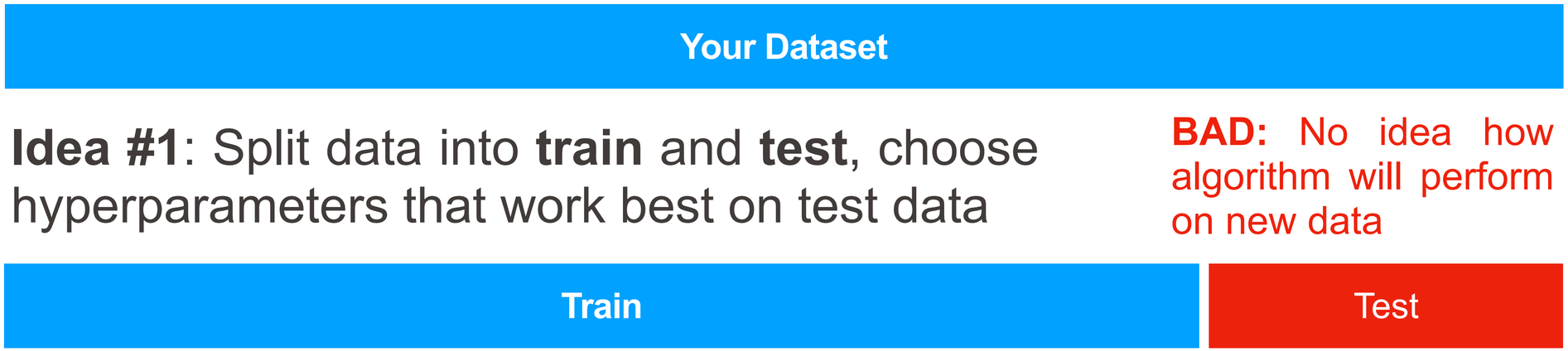
- Next - how do we set the regularization parameter λ ?

Train, validate, test in ML

Setting hyperparameters: example, the regulariser

Train, validate, test in ML

Setting hyperparameters: example, the regulariser



Idea #1: Split data into **train** and **test**, choose hyperparameters that work best on test data

BAD: No idea how algorithm will perform on new data

Train, validate, test in ML

Setting hyperparameters

Idea #1: Split data into **train** and **test**, choose hyperparameters that work best on test data

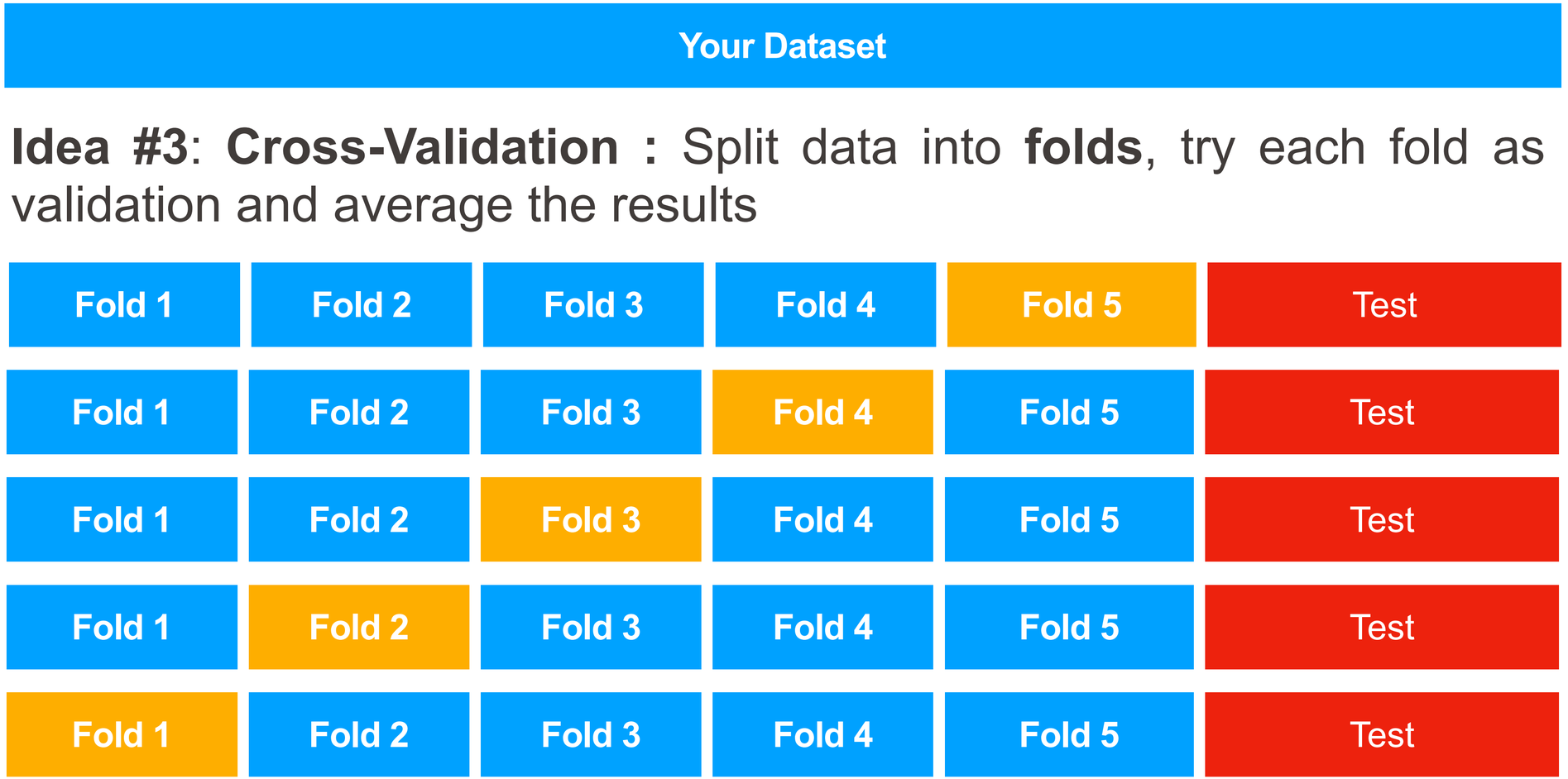
Problem: No idea how algorithm will perform on new data

Idea #2: Split data into **train, val** and **test**, choose hyperparameters on val and evaluate on test

Better!

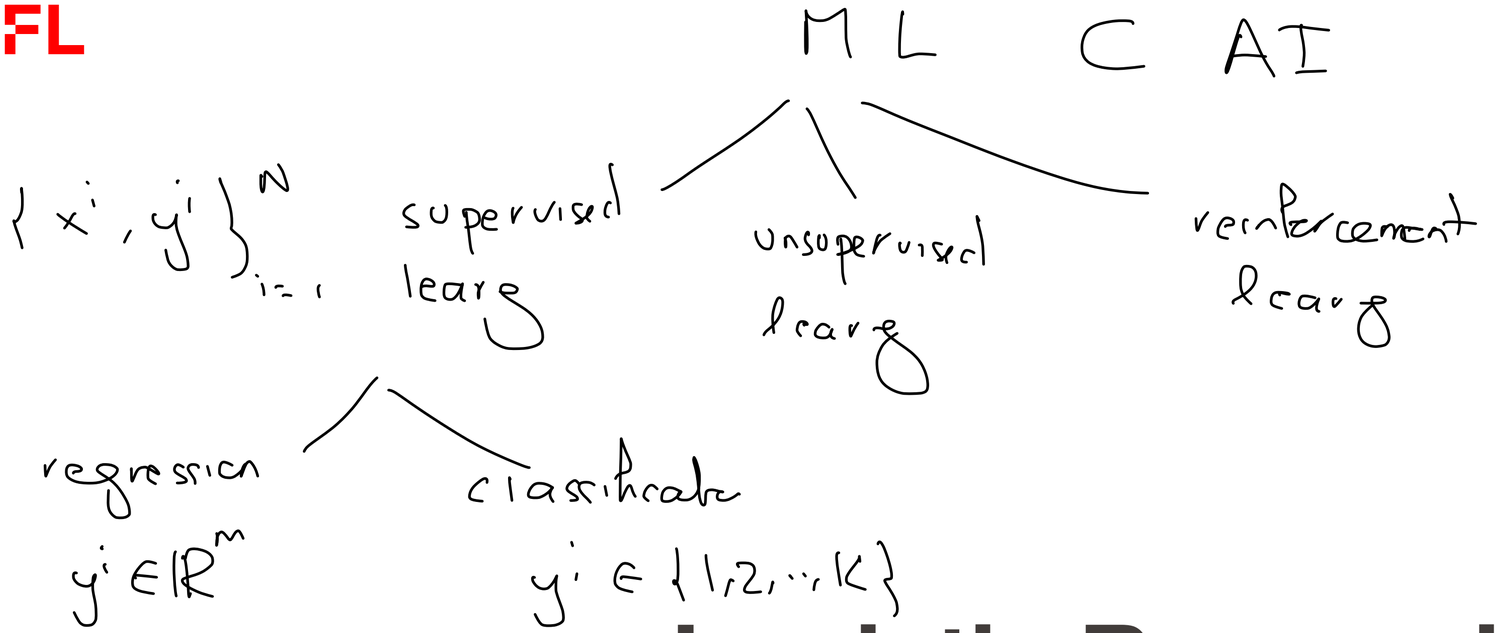
Train, validate, test in ML

Setting hyperparameters



Idea #3: **Cross-Validation :** Split data into **folds**, try each fold as validation and average the results

Useful for small datasets



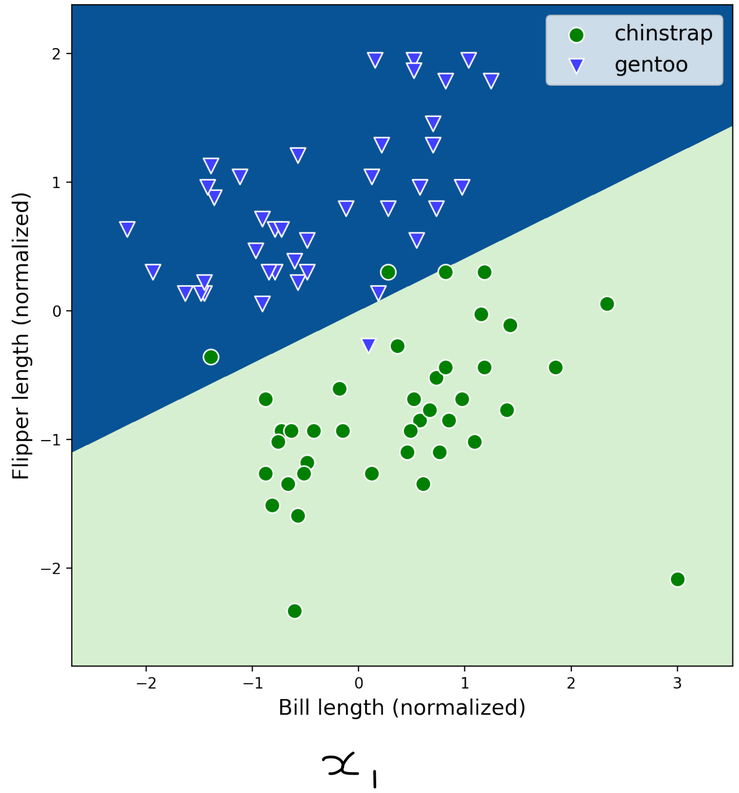
Logistic Regression

is a classification technique.

Classification

Palmer Penguins

	species	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
0	Chinstrap	49.0	19.5	210.0	3950.0
1	Chinstrap	50.9	19.1	196.0	3550.0
2	Gentoo	42.7	13.7	208.0	3950.0
3	Chinstrap	43.5	18.1	202.0	3400.0
4	Chinstrap	49.8	17.3	198.0	3675.0



Based on dataset available here: https://www.kaggle.com/code/parulpandey/penguin-dataset-the-new-iris

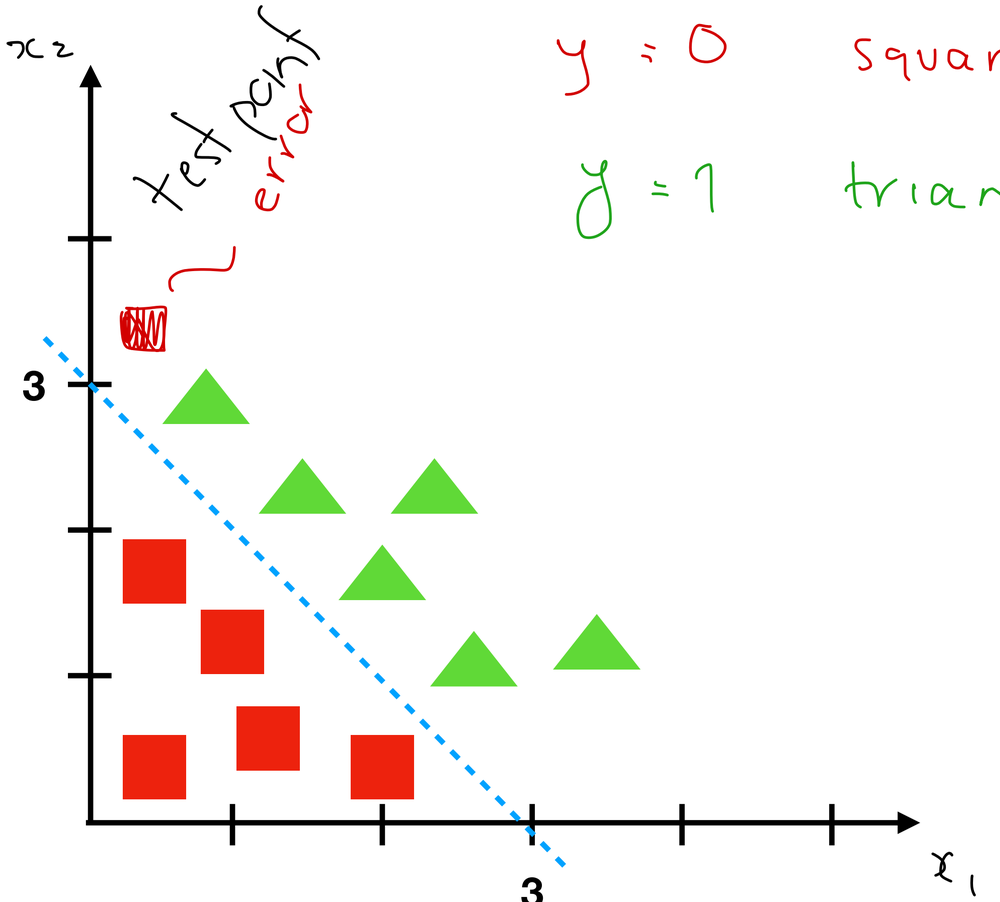
EPFL

Logistic Regression (binary classification)

Characterize a classifier based on a linear model

Goal: find a line/hyperplane separating the classes

$$z = b + w_1 x_1 + w_2 x_2 + \cdots + w_d x_d \quad \text{or} \quad f_w(x) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases}$$



$y = 0$ square

$y = 1$ triangle

Ex: $\mathbf{w} = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} b \\ w_1 \\ w_2 \end{bmatrix}$

Predict triangle if $-3 + x_1 + x_2 \geq 0$

Predict square else

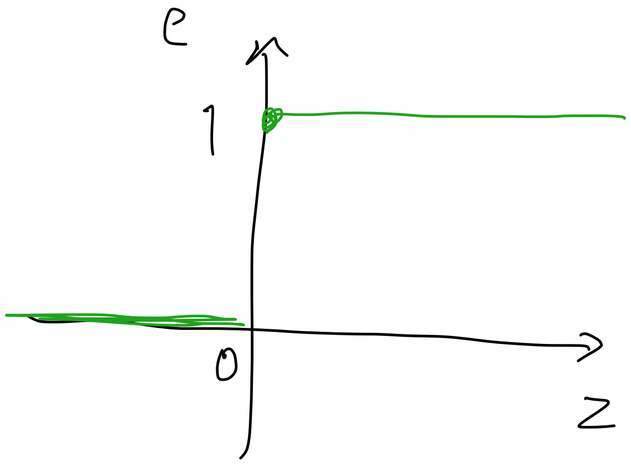
Defining loss function for classification

Consider classifier : class 1 if $z \geq 0$

class 0 else

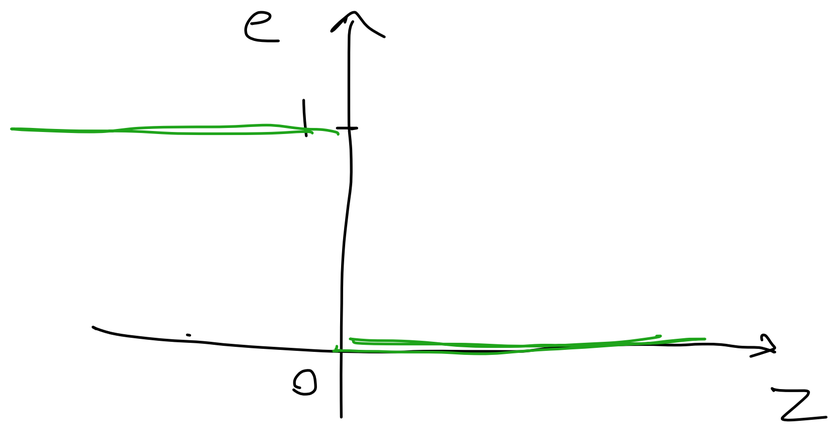
$(z = b + w_1 x_1 + \ldots + w_d x_d)$

Error : $|$predicted label − true label$|$



true label is

$y = 0$



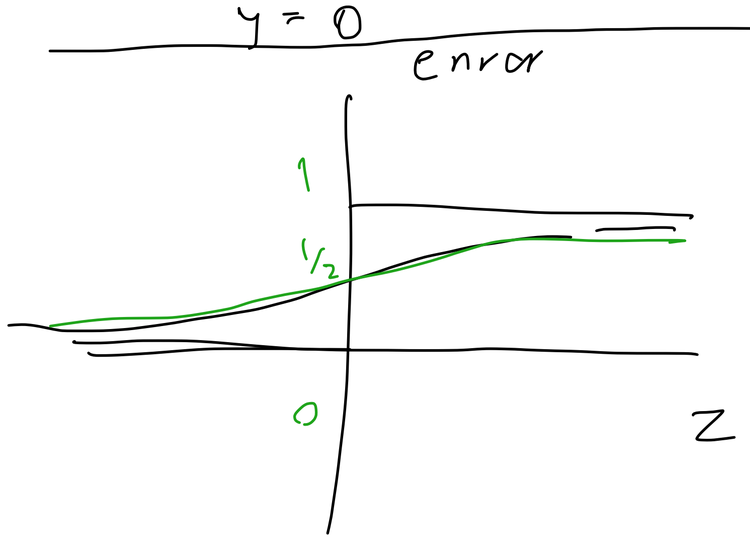
true label is

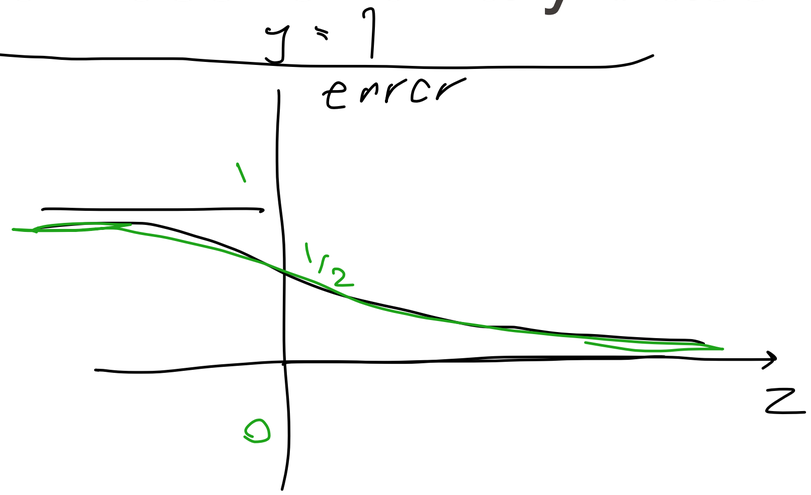
$y = 1$

Given x^i, we can find $\hat{y}^i$ & compute the error

Logistic loss function

A differentiable and convex loss for binary classification

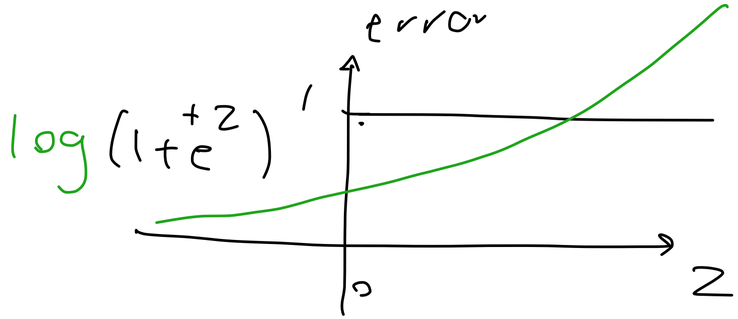


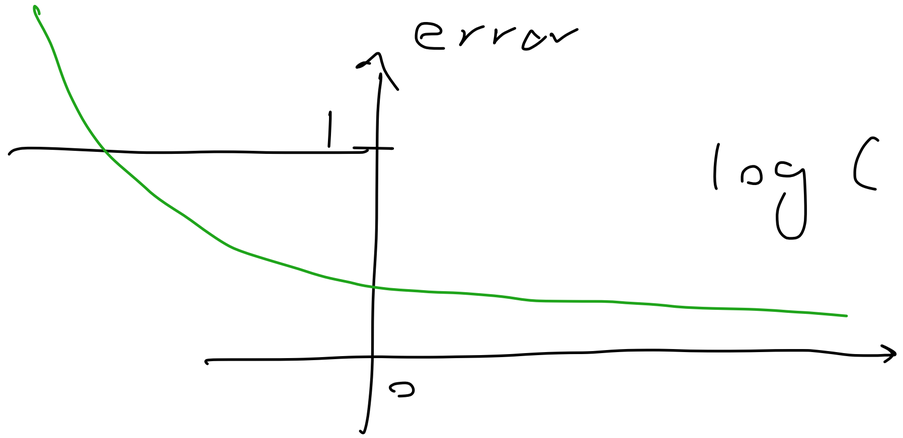


$$\sigma(z) = \frac{1}{1+e^{-z}}$$ sigmoid function

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

non-convex





Read more about logistic function/verify your gradient computation here: https://en.wikipedia.org/wiki/Logistic_function

Training - Minimizing the logistic loss function

$\{(x^i, y^i)\}_{i=1}^N \qquad y^i \in \{0,1\}$

$z = b + w_1 x_1 + \dots + w_d x$

$$\hat{y}^i = \begin{cases} 1 & z(x^i) \geq 0 \\ 0 & \text{else} \end{cases}$$

$$J(b,w) = \frac{1}{N} \sum_{i=1}^{N} y^i \log\left(1 + e^{-z^i}\right) + (1-y^i) \log\left(1 + e^{z^i}\right)$$

logistic loss function, it used for classification ($y^i \in \{0,1\}$)
(binary cross entropy loss)

convex function ..

↳ gradient descent to update the parameters: $w = \begin{bmatrix} b \\ w_1 \\ \vdots \\ w_d \end{bmatrix} \in \mathbb{R}^{d+1}$ (w_0)

$$w^{(t+1)} = w^{(t)} - \alpha \nabla_w J\left(w^{(t+1)}\right)$$

Exercise

Consider the sigmoid function $\sigma : \mathbb{R} \to (0,1)$, $\sigma(z) = \frac{1}{1+e^{-z}}$. Compute $\frac{d\sigma(z)}{dz}$. Use the chain rule to compute $\frac{d\sigma(z(w))}{dw}$, where $z = w_0 + w_1 x_1 + \ldots + w_d x_d$ and. $w = (w_0, w_1, \ldots, w_d)$

Compute the gradient of the binary cross-entropy loss function with respect to the parameters $w = (w_0, w_1, \ldots, w_d)$

You can check your answers with the notes in the python exercises this week.

Probabilistic interpretation of logistic function

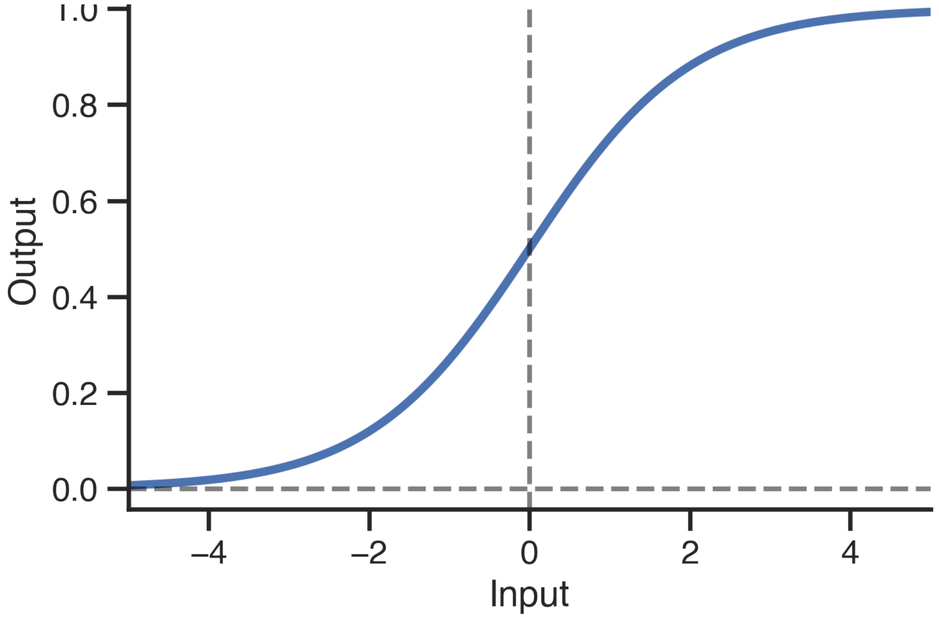
Consider the regression

- $z^i = \mathbf{w}^T \boldsymbol{x}^i + b$

Followed by the logistic function $\hat{y}^i = \dfrac{1}{1 + e^{-(\mathbf{w}^T \boldsymbol{x}^i + b)}}$

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

$$\sigma : \mathbb{R} \to (0,1)$$



Rather than saying

$z \geq 0$ class 1

else class 2, we can

say $\hat{y}^i = \dfrac{1}{1+e^{-z^i}}$ is probably of class 1

- $\hat{y}^i$ = estimated probability of class 1 on input $\boldsymbol{x}^i$
- $\hat{y}^i = P(y^i = 1 \,|\, \boldsymbol{x}^i; \mathbf{w})$ estimated probably of class 1 given $\boldsymbol{x}^i$, parameterized by $\mathbf{w}$
- $1 - \hat{y}^i = P(y^i = 0 \,|\, \boldsymbol{x}^i; \mathbf{w})$

Probability distributions background

$S = \{1, 2, \dots, n\}$, a probability distribution on S is a function $p : S \to \mathbb{R}$ s.t. $p(i) \geq 0 \quad \forall i \in S$ and $\sum_{i=1}^{n} p(i) = 1$, we can equivalently represent this function as a vector $p = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{bmatrix}$, where $p_i = p(i)$

Cross entropy of a distribution relative to another

probability

Let $p, q : \{1, 2, \dots, n\} \to \mathbb{R}$ two probability distributions

Then, the cross-entropy of q with respect to distribution p is defined as $-\sum_{j=1}^{n} p(j) \log(q(j))$

Our estimated distribution is $\hat{p}$: probability of prediction label y given x.

true distribution is p

Cross-entropy of $\hat{p}$ relative to p

$$-\left[p(y=1|x)\log(\hat{p}(y=1|x)) + p(y=0|x)\log(\hat{p}(y=1|x))\right]$$

Logistic loss

Interpretation as binary cross-entropy loss

By interpreting the logistic function as probability of a label,

$z^i = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_d x_d$

$\hat{y}^i = \frac{1}{1+e^{-z_i}}, \quad 1-\hat{y}^i = \frac{1}{1+e^{z_i}}$

true distribution : $p(y|x)$ unavailable .. all we have is data $\{x^i, y^i\}_{i=1}^N$

The cross-entropy between the true (unknown) distribution and the estimated distribution

$$-\frac{1}{N}\left(\sum_{i=1}^{N} y^i \log \frac{1}{1+e^{-z_i}} + (1-y^i)\log \frac{1}{1+e^{z_i}}\right)$$

The cross entropy loss below, is the same as the logistic loss

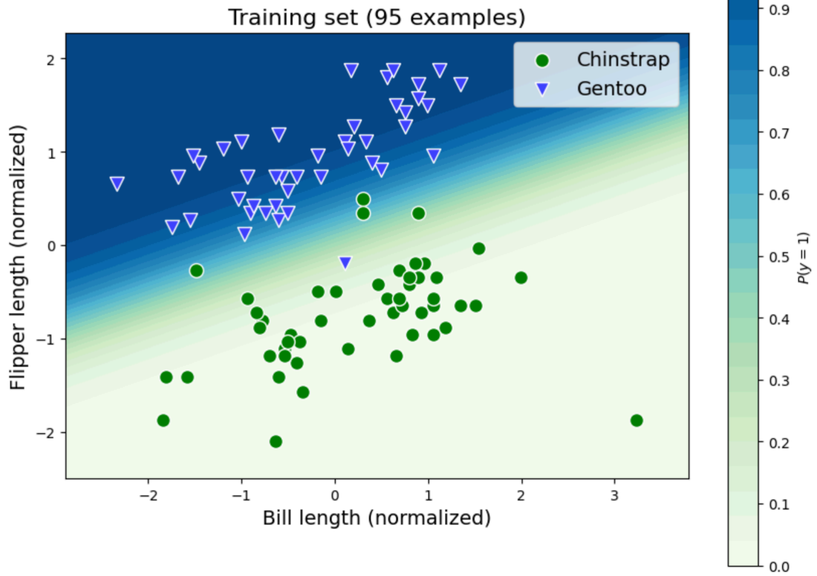
$$= \frac{1}{N}\sum_{i=1}^{N} y^i \log\left(1+e^{-z^i}\right) + (1-y^i)\log\left(1+e^{z^i}\right)$$

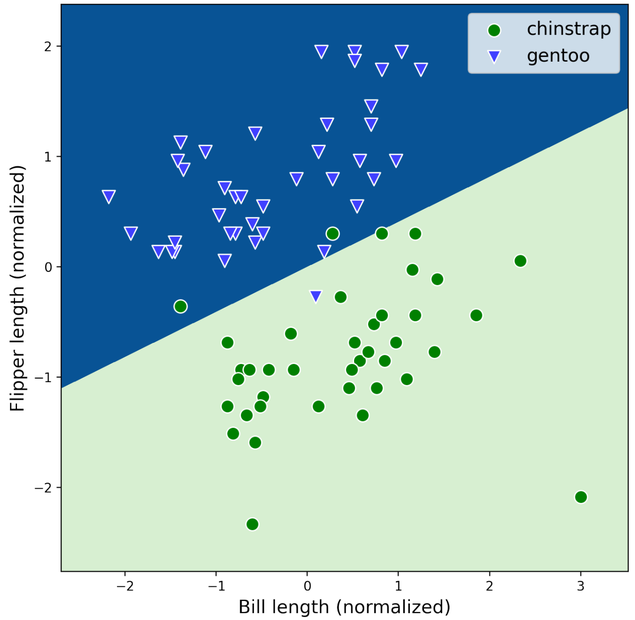
same as our logistic loss.

Logistic regression output

Palmer Penguins

	species	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
0	Chinstrap	49.0	19.5	210.0	3950.0
1	Chinstrap	50.9	19.1	196.0	3550.0
2	Gentoo	42.7	13.7	208.0	3950.0
3	Chinstrap	43.5	18.1	202.0	3400.0
4	Chinstrap	49.8	17.3	198.0	3675.0

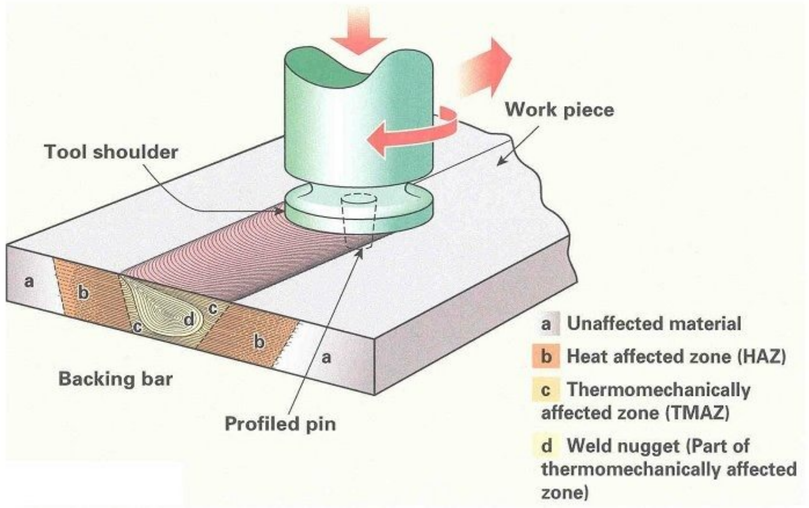


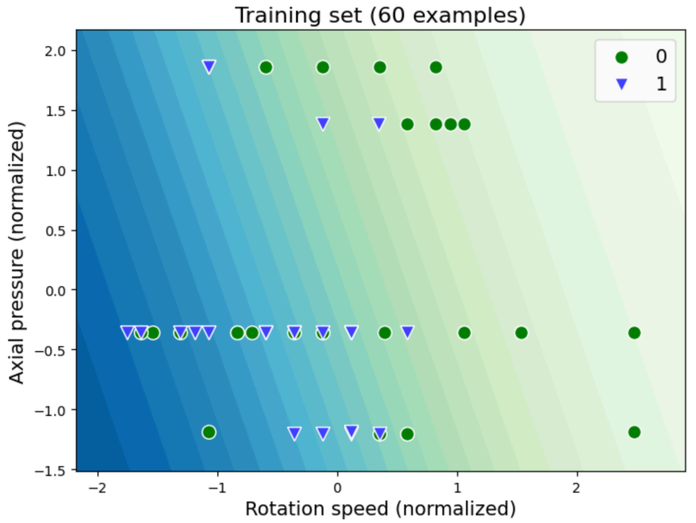


Based on dataset available here: https://www.kaggle.com/code/parulpandey/penguin-dataset-the-new-iris

Logistic regression exercise this week

- **Dataset 1:** Void Formation in Welding, based on the paper
- **Goal:** formation of voids in friction stir welding as a function of the operation conditions
  - Tool rotational speed, axial pressure
  - The label: void or not void





- **Dataset 2:** discriminate between sonar signals bounced off a mine (metal cylinder) and those bounced off a roughly cylindrical rock
- **Goal:** predict whether the object is mine or rock based on
  - The features (60 of them) are the energy within a particular frequency band, integrated over a certain period of time
  - The label: rock/mine

Note on implementation
Data normalization

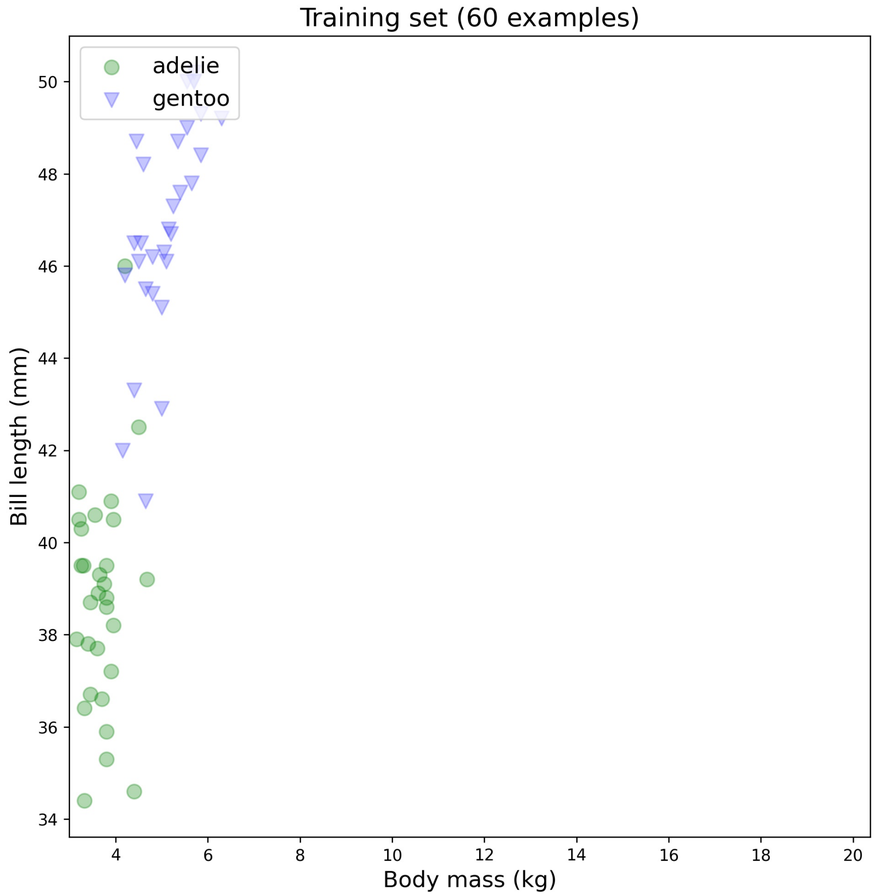
Data normalization / feature scaling: Normalize features (bring them all to the same scale)

Crucial step in preprocessing:

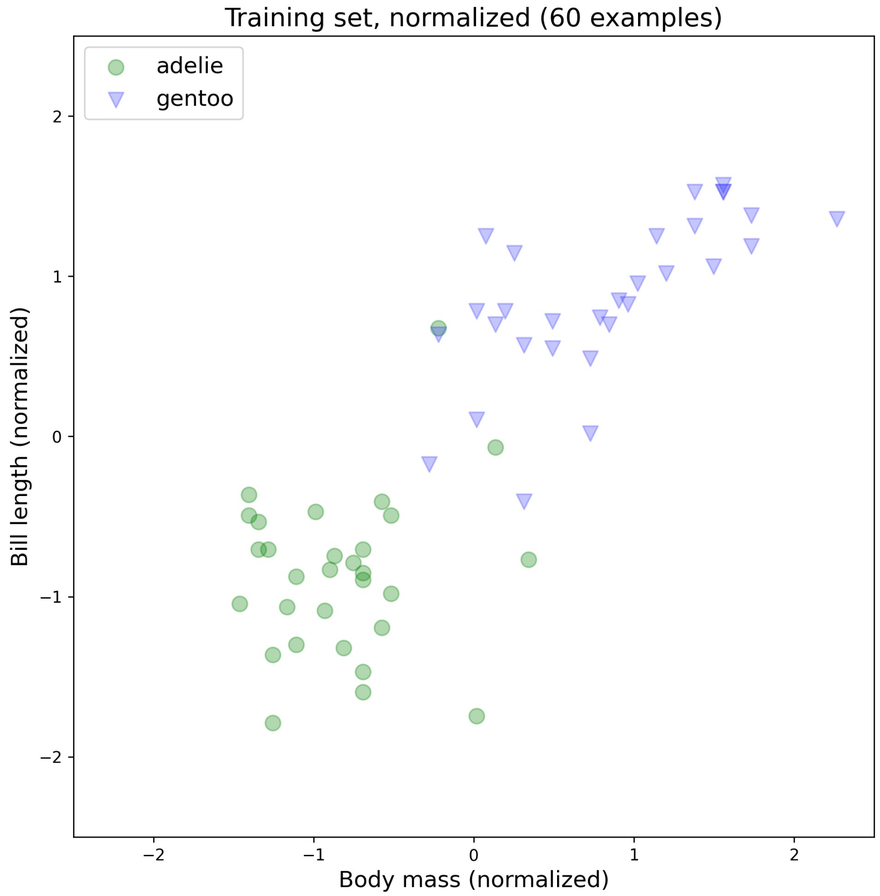
- Many classifiers (e.g. KNN that we will see in next lecture) rely on distance metrics
- Gradient descent will converge faster
- Coefficients are penalized appropriately (in the case where regularization is applied)

Numerical features

Data normalization - Example



Normalization



Performance metrics for binary classification

Confusion matrix

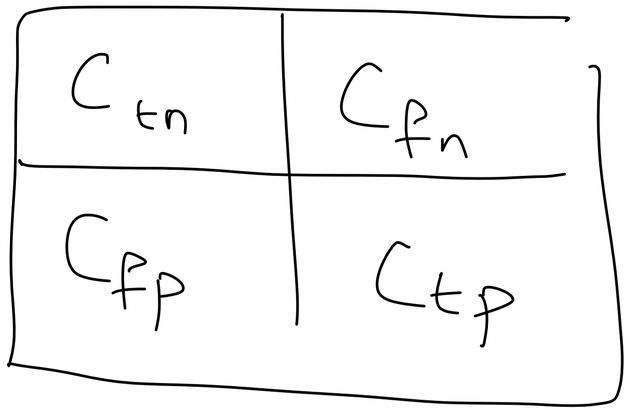
false positives : true label is 0, we predicted 1

false negatives : " " " 1, " " 0

true positives : " " " 1, " " 1

true negatives " " " 0, " " 0

Confusion matrix



Performance metrics for binary classification

Accuracy, error rate, recall

$$\text{accuracy} = \frac{C_{tn} + C_{tp}}{N}$$

$$\text{error rate} = \frac{C_{fn} + C_{fp}}{N}$$

$$\text{recall} = \frac{C_{tp}}{C_{tp} + C_{fn}}$$

Exercise

Performance metric for binary classification

- We have used two approaches to train classifiers for spam email detection: “non-spam” (class 0) and “spam” (class 1)
- Our test set has1000 emails, 900 of which were non-spam
- Approach 1: classified all data as non-spam
- Approach 2: classified 850 of non-spam emails as non-spam and 50 of spam emails as spam
- Write the confusion matrix of each approach
- Compute the error rate, accuracy and recall of each algorithm
- What do you conclude?

Outline

- Overfitting/underfitting and regularization
  - Connections to sensitivity and Lipschitz constant of the predictor
- Logistic regression
  - differentiable and convex loss function
  - Probabilistic interpretation
  - Performance metric different than the loss function

- Announcements:
  - Exercise hour: logistic regression (notice data normalization - more about it next week)
  - Check moodle for past year's exams and quizzes