FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Linear Regression, Logistic Regression & Naïve Bayes

04 – ML MODELS

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Agenda Today

- Linear Regression
  - Gradient Descent
- Logistic Regression
- Naïve Bayes

Recap: Bias – Variance tradeoff

> What is the difference between Bias and Variance?
> What is the difference between under- and over-fitting?
> How can we overcome these issues?

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Recap: Canonical Types // output size // un/supervised?

How would you tackle the following tasks? What *canonical problem type* are they? What is their output space *size*?

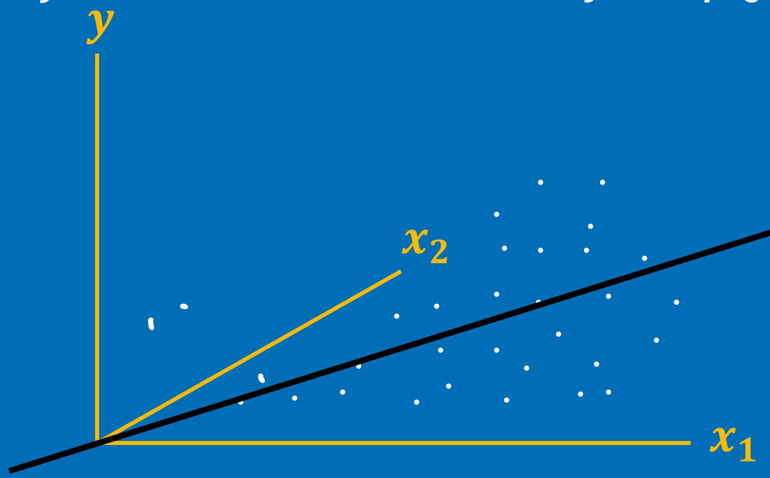
> Assign one of 36 POS tags to each word of a given sentence.
> Decide if an email is a spam.
> Determine if a comment is hateful, toxic, threat, insult, or a combination.
> Automatically extract and assign a set of 10 topics from 1000 documents.
> Given a comment about a product, predict the score the user gave.

Linear Regression

> A **Supervised Learning** method

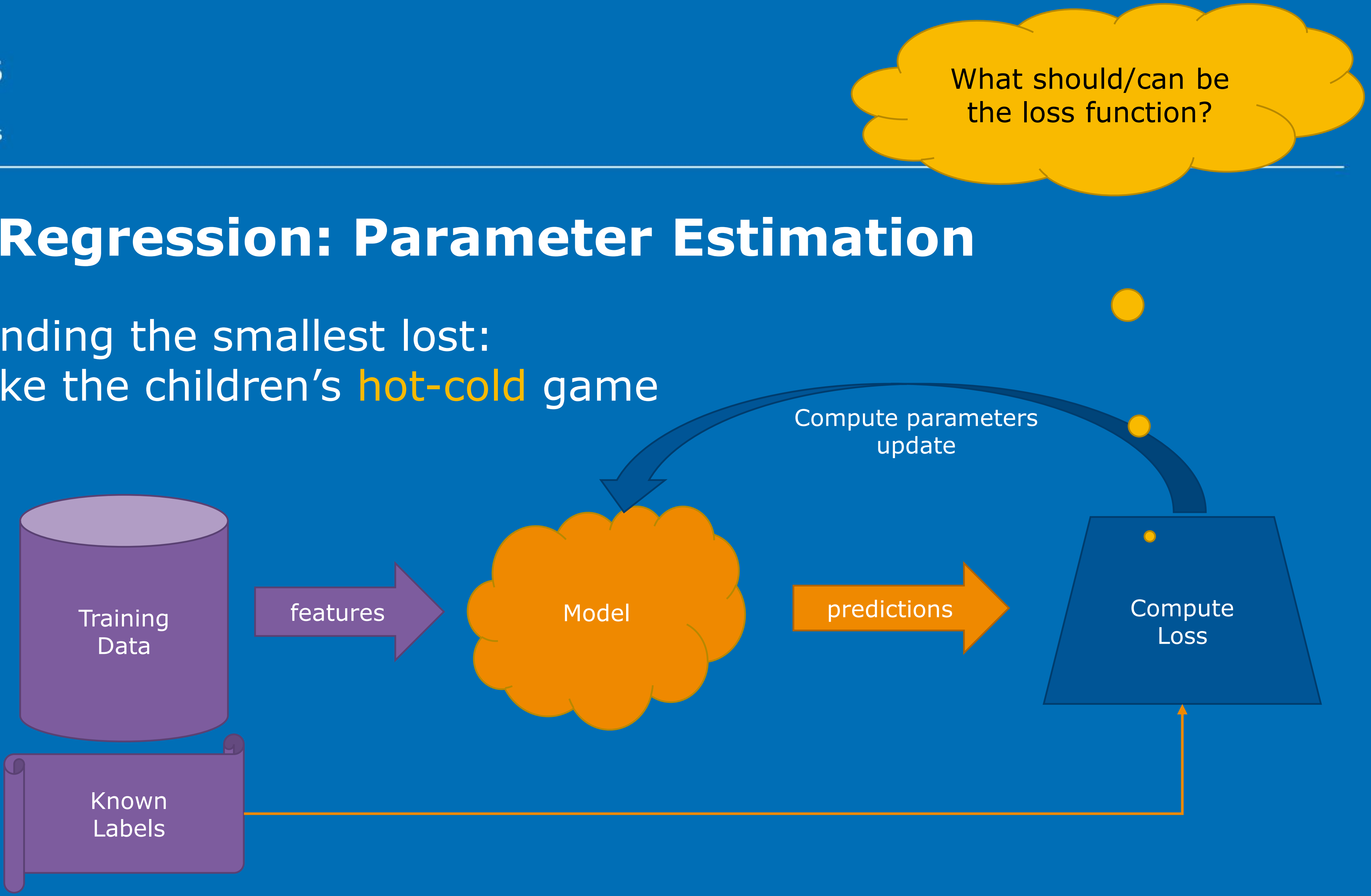
> Predicting a value $\hat{y} \in \mathbb{R}$ from m features $x \in \mathbb{R}^m$ by learning a linear function: $f(x) = y$

> f is of the form: $\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \ldots = \beta_0 + x^T \beta$



Linear Regression: Parameter Estimation

Finding the smallest lost:
Like the children's hot-cold game

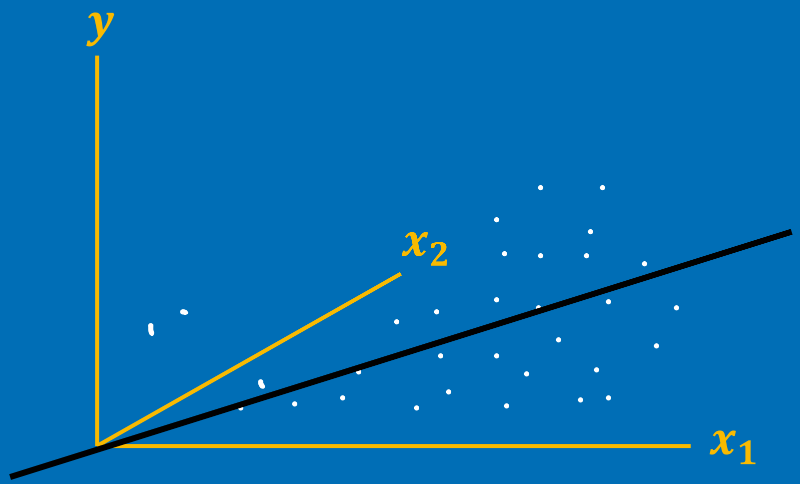


FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Linear Regression: Parameter Estimation

We can estimate the best parameters $\langle \beta_0, \beta_1 \rangle$ by minimizing a *loss* function.

A possible *loss*: the average of squared errors (MSE) between the true values and the predictions with these parameters:



$$\hat{\theta} = \arg \min_{\theta = \langle \beta_0, \beta_1 \rangle} \frac{1}{2n} \sum_{i=1}^{n} \left(y_i - \left(\beta_0 + x_i^T \beta \right) \right)^2$$

Linear Regression: Parameter Estimation

Simplified case with 2 parameters:

$$\hat{y} = \beta_0 + \beta_1 x_1$$

How to find the best parameters $\langle \beta_0, \beta_1 \rangle$?

1. Initialize with random values
2. Calculate the Loss: e.g., $\sum_{i=1}^{n} \left(y_{true} - y_{pred}\right)^2$
3. Calculate the necessary parameters change.
4. Repeat, until the average loss is very small, or the parameter change is minimal.

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Linear Regression: Parameter Estimation

Simplified case with 2 parameters:

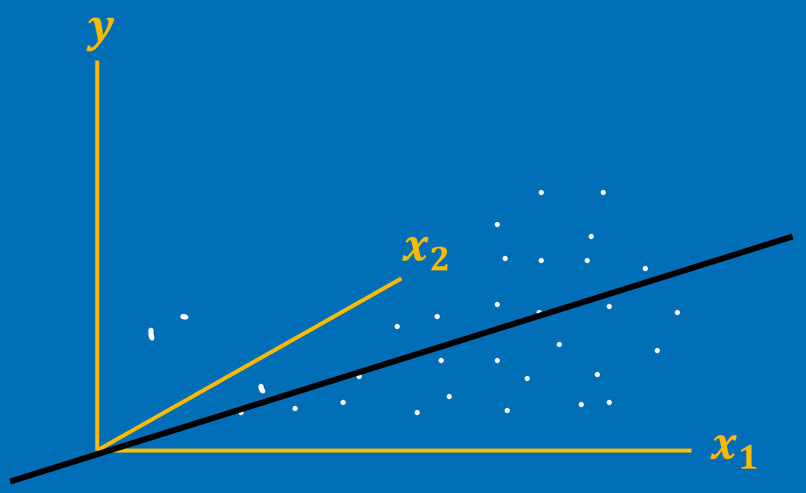
$$\hat{y} = \beta_0 + \beta_1 x_1$$

How to find the best parameters $\langle \beta_0, \beta_1 \rangle$?

1. Initialize with random values
2. Calculate the Loss: e.g., $\sum_{i=1}^{n} \left(y_{true} - y_{pred}\right)^2$
3. Calculate the necessary parameters change.
4. Repeat, until the average loss is very small, or the parameter change is minimal.

Linear Regression: Least Mean Squares (LMS)

Parameter Estimation is done through calculating the cost(=loss) function *partial derivatives* with respect to each β and finding its global minima with a method called:
Gradient Descent



$$\hat{\theta} = \arg \min_{\theta=\langle\beta_0,\beta_1\rangle} \frac{1}{2n} \sum_{i=1}^{n} \left(y_i - \left(\beta_0 + x_i^T \beta\right)\right)^2$$

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

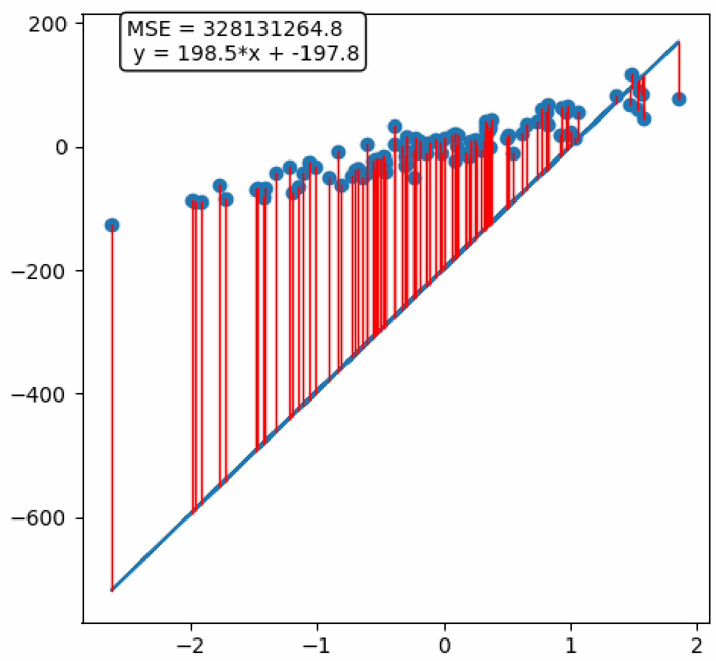
Optimizing Linear Regression with Gradient Descent

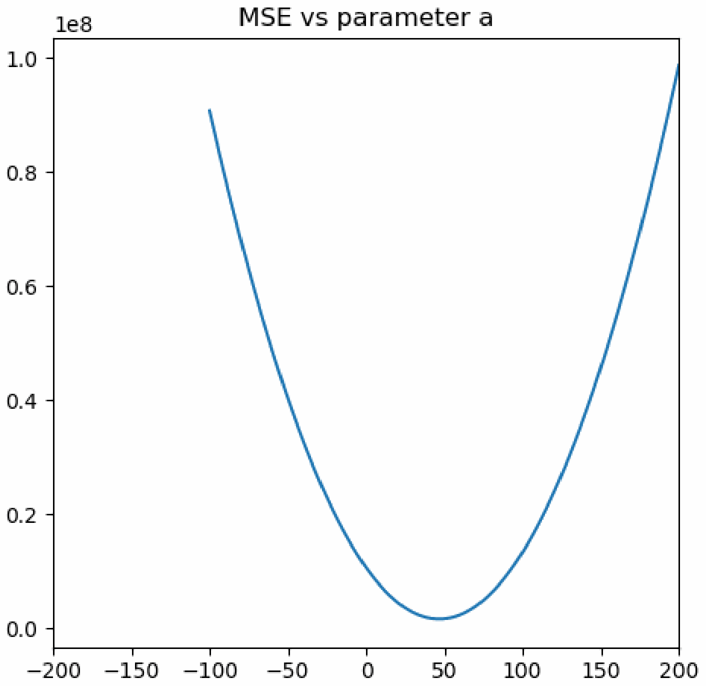
- $\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2$ …
- Calculating partial derivatives **(gradient)**
- Finding the slope – which direction minimizes **(descent)** the gradients?
  And by how much?
  (Multiplied by some fraction - Learning Rate)

Gradient Descent

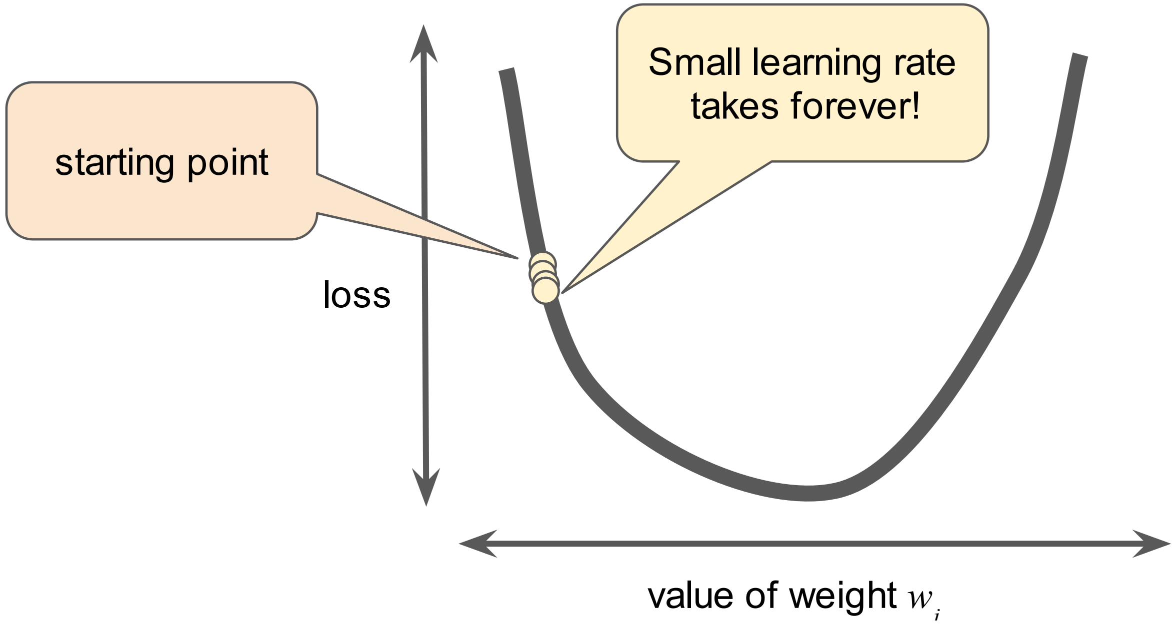
> Minimizing the partial derivatives (like in high school) – to find the global minimum

> Often is multiplied by some **learning-rate**
if too big – jumps too far
if too small – takes too long



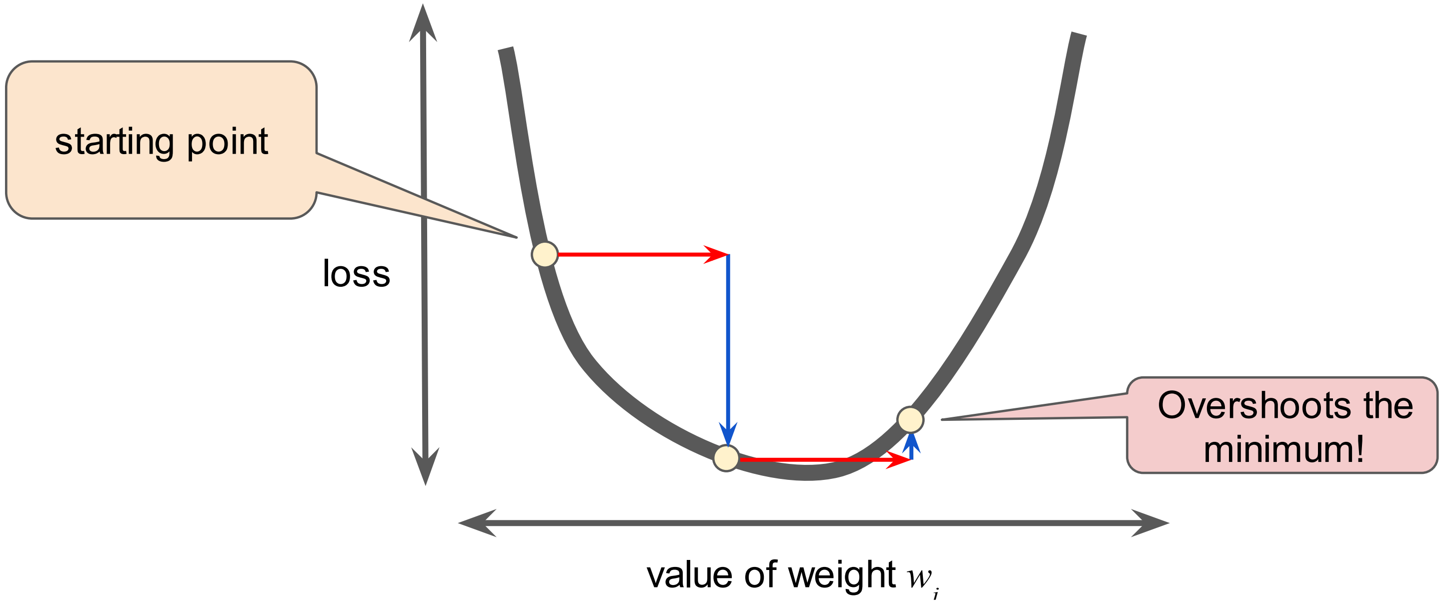


Learning Rate – too small

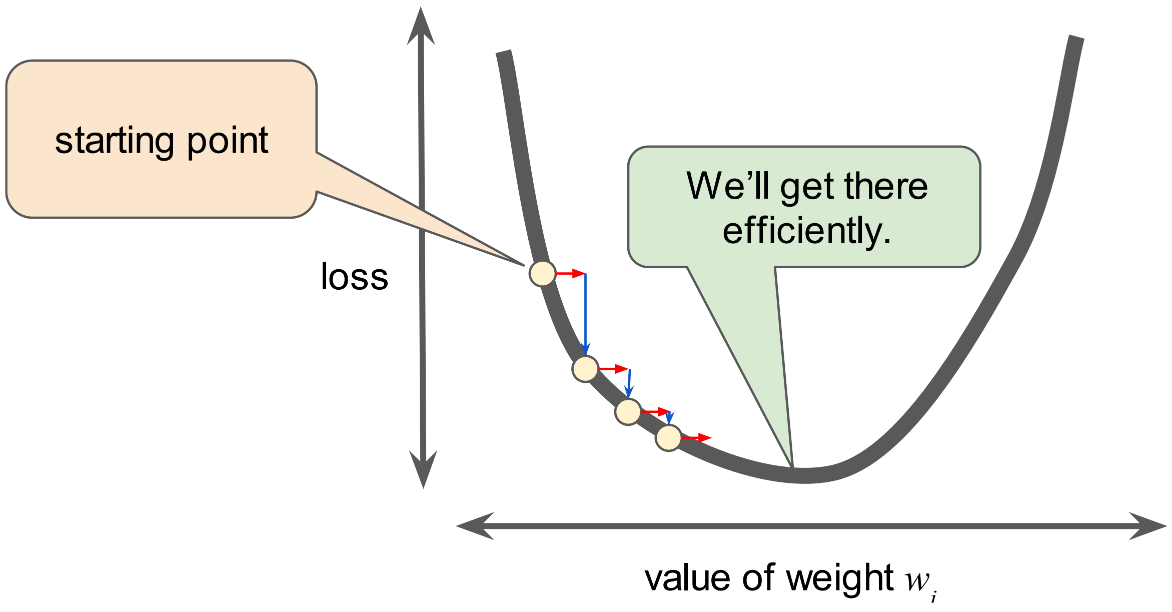


FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Learning Rate – too large



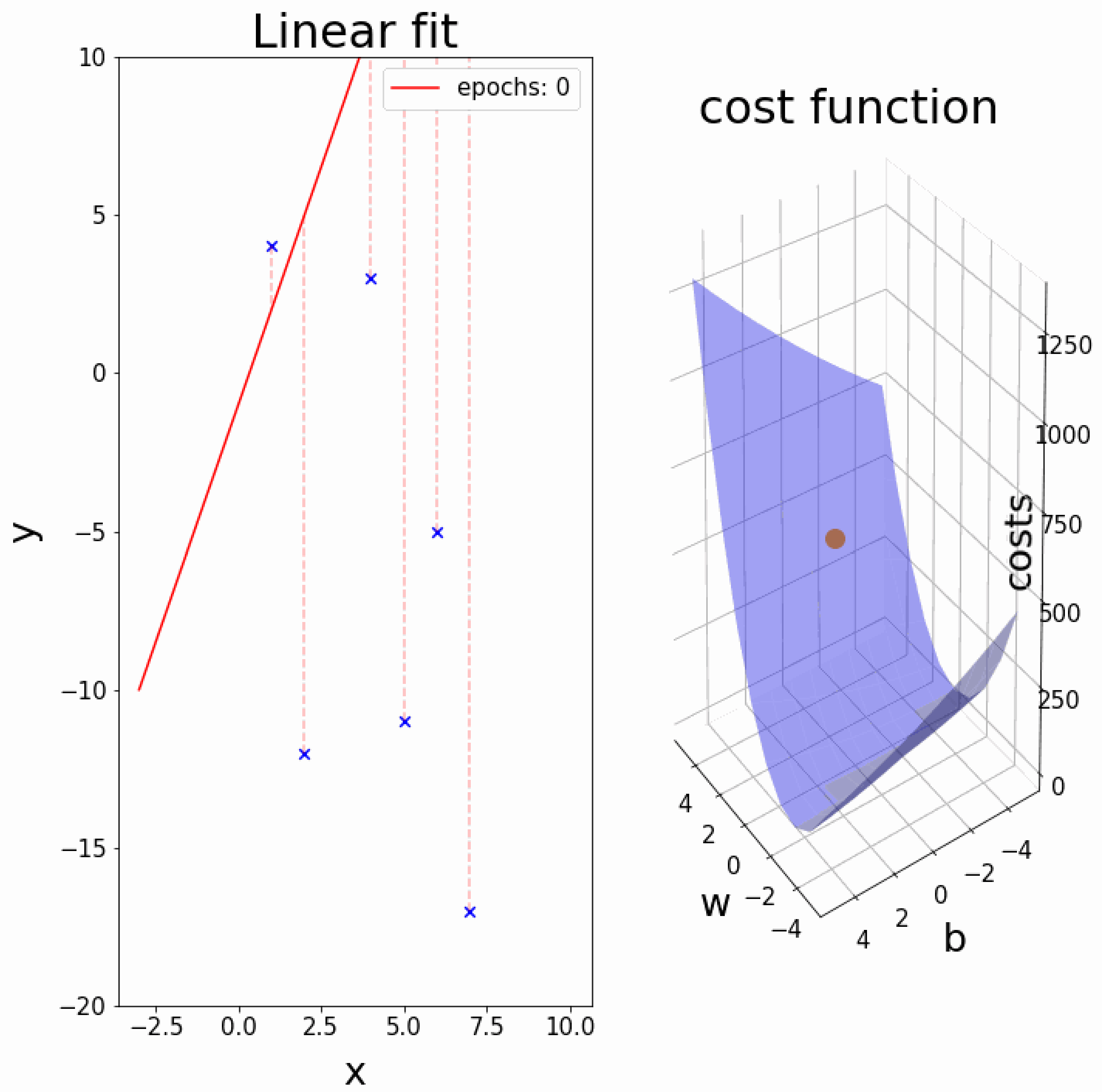
Learning Rate – just right



Two-dimensional

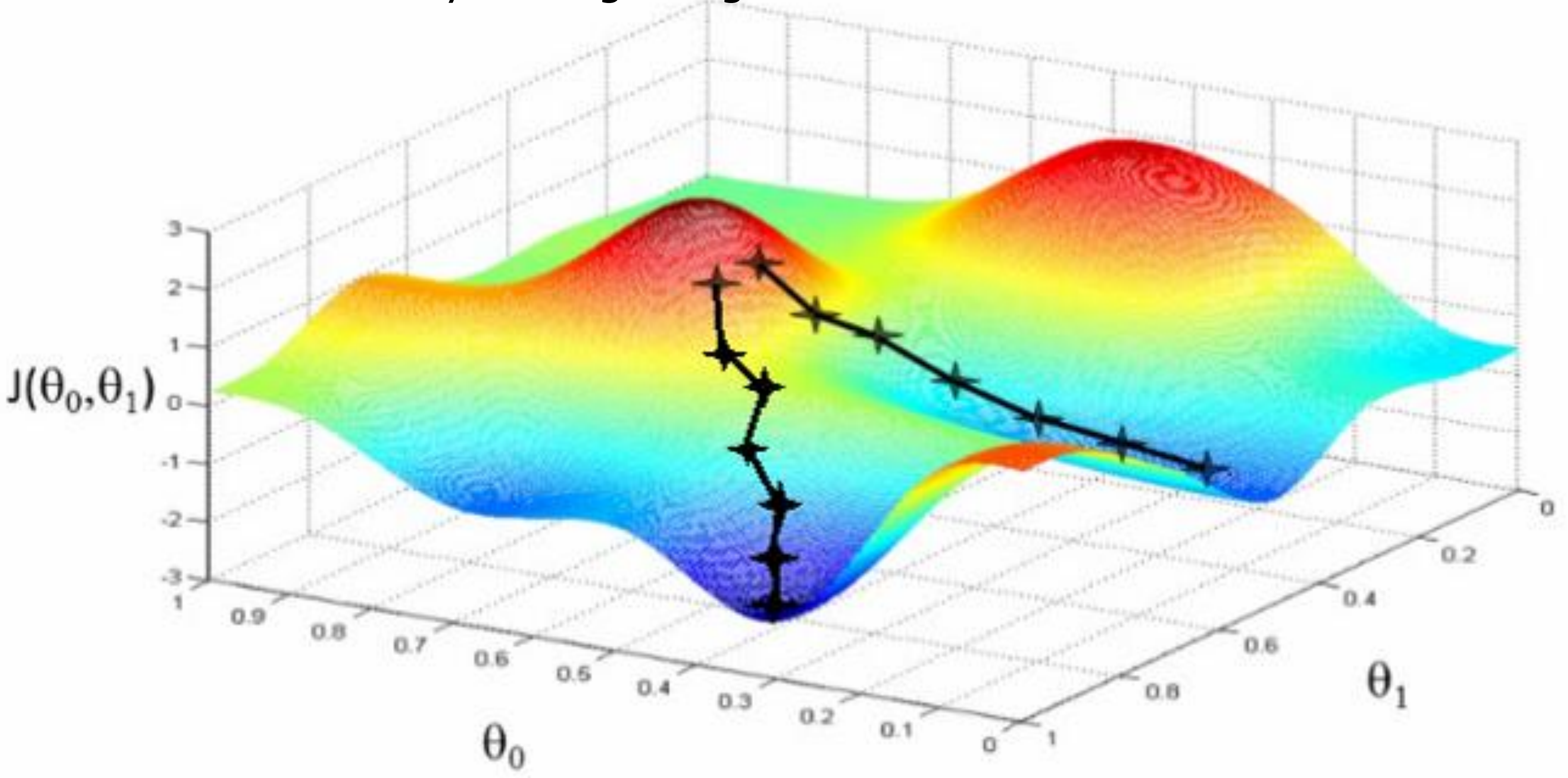
> $\hat{y} = \beta_0 + \beta_1 x$

> Minimizing both - β_0 & β_1

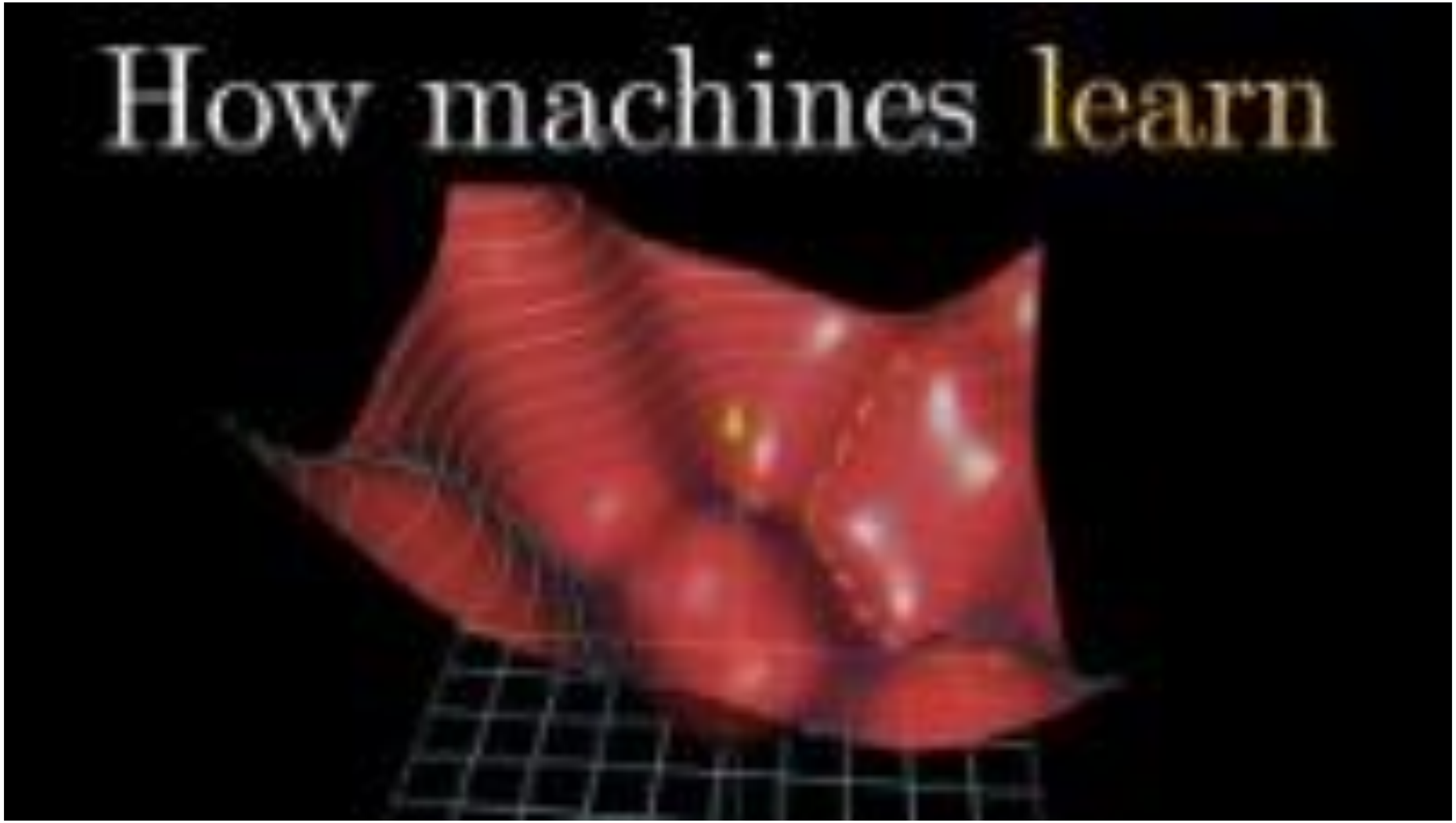


Gradient Descent

Minimizes the cost function by finding the global minima of the derivatives



Gradient Descent: Recommended Video



Back to Linear Regression

- Calculates the best linear fit of the given features
- Assumes the features are **independent (!)**
- Requires numerical features
  - Which features would you craft for the following problem: Given an essay, predict the student’s score (0-100)

Linear Regression - Regularization

> ML training can cause overfitting.
  > Q: What would overfitting look like for **Linear** regression?

> How can we prevent it?

Regularization!

$$\hat{\theta} = \arg \min_{\theta=\langle\beta_0,\beta_1\rangle} \frac{1}{2n} \sum_{i=1}^{n} \left(y_i - \left(\beta_0 + x_i^T \beta\right)\right)^2$$

Linear Regression - Regularization

> ML training can cause overfitting.
> > What would overfitting look like for **Linear** regression?

> How can we prevent it?

L_2 Regularization: Euclidean Distance

$$\hat{\theta} = \arg \min_{\theta=\langle \beta_0, \beta_1 \rangle} \frac{1}{2n} \sum_{i=1}^{n} \left(y_i - \left(\beta_0 + x_i^T \beta \right) \right)^2 + \frac{\lambda}{2} \sqrt{\sum_{j=1}^{m} \beta_j^2}$$

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Linear Regression - Regularization

> ML training can cause overfitting.
>
> > What would overfitting look like for **Linear** regression?

> How can we prevent it?

L_1 Regularization: Manhattan (Taxicab) Distance

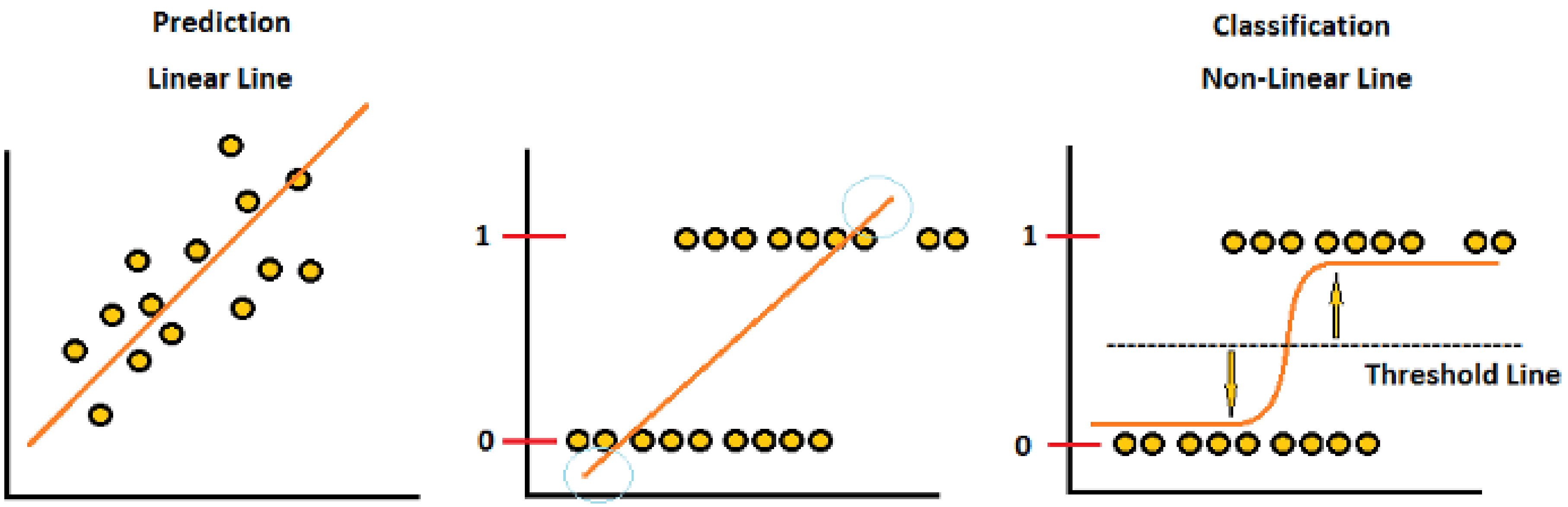
$$\hat{\theta} = \arg\min_{\theta=\langle\beta_0,\beta_1\rangle} \frac{1}{2n}\sum_{i=1}^{n}\left(y_i - \left(\beta_0 + x_i^T\beta\right)\right)^2 + \lambda\sum_{j=1}^{m}\left|\beta_j\right|$$

From Linear to Logistic

Classification using
Logistic Regression

Binary Logistic Regression

> Linear Regression: Predict an essay’s **score** (e.g., readability)
> Logistic Regression: Predict a **probability** (Readable or not)



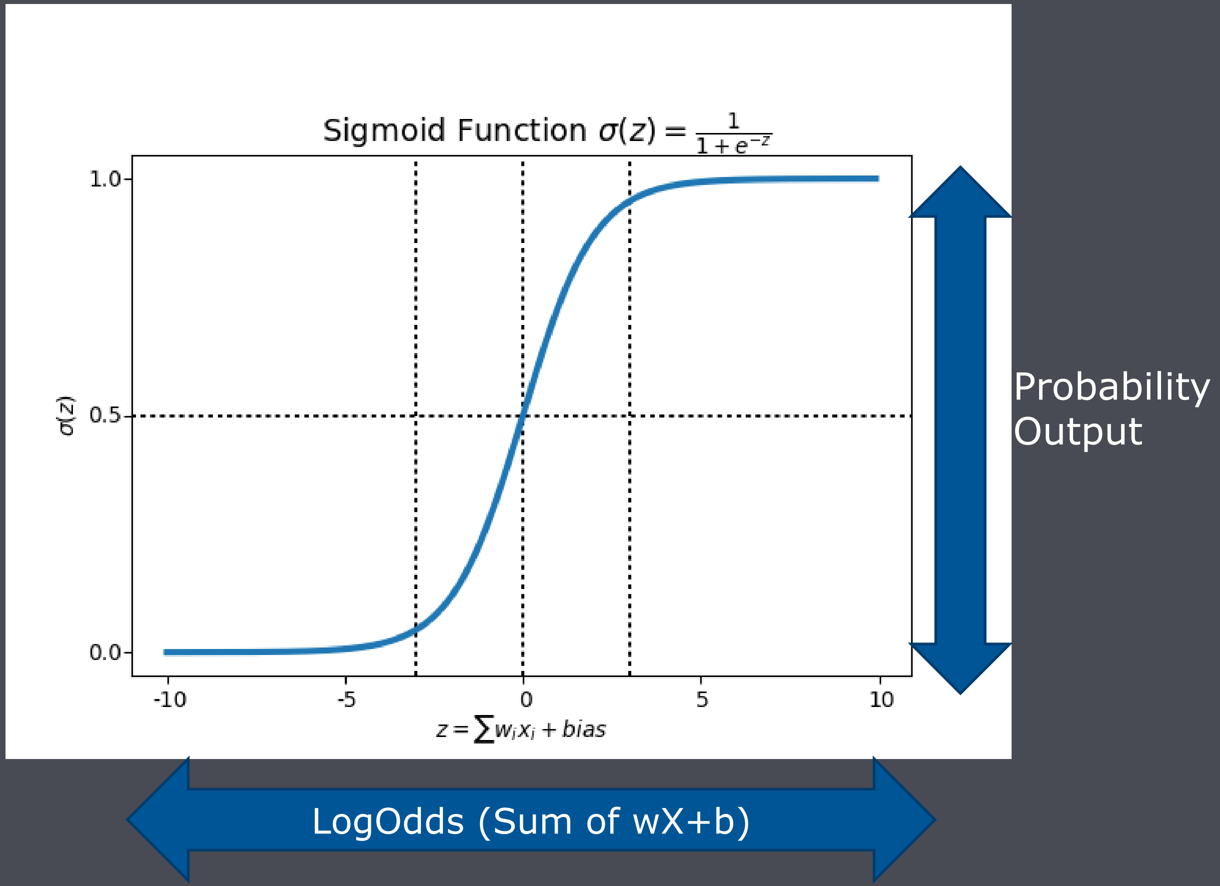
Linear Logistic Regression

> Gives a **probability** estimation

> By wrapping the linear function in a non-linear function.
E.g., *sigmoid function*:

$$y' = \frac{1}{1 + e^{-(w^T x + b)}}$$

> Squishing the linear output through sigmoid



Linear Logistic Regression – Sigmoid Function

- Probability vs Odds.
- Probability: 0…1

- Odds: $\frac{P}{1-P}$
- Logistic Function:

$$\ln\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 x$$

$$\frac{P}{1-P} = e^{\beta_0+\beta_1 x}$$

$$P = \frac{1}{1+e^{-(\beta_0+\beta_1 x)}}$$

Y	*1*	*0*
Pr(Y=1)	*P*	*1- P*

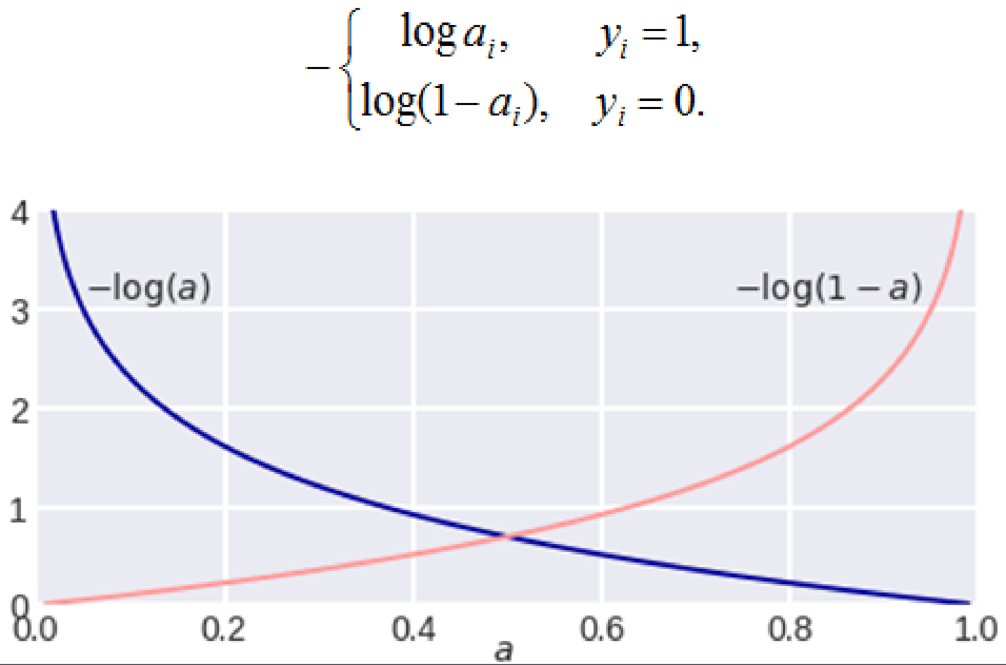
**P= Success, 1-P = Failure*

Linear Logistic Regression - *Loss* Function

- In Linear Regression we used **Mean Squared Error (MSE).**
- In Logistic Regression we need a different *loss*:

$$Log\ Loss = \sum_{(x,y)\in D} -y\log(\hat{y}) - (1-y)\log(1-\hat{y})$$

- As we reach closer to the side of the bars, the probability gets high quickly.



$(x, y) \in D$ – data and its labels
y – the true label (between 0 and 1)
y’ – the predicted label (between 0 and 1)

Regularization

> Regularization is very important in logistic regression modeling.

> Without it, the loss would go towards 0 in high dimensions (due to the asymptotic nature of logistic regression)

> Strategies to dampen model complexity:

1. L_2 regularization.
2. Early stopping

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Example: Sentiment Classification of movie reviews

> Problem definition: Estimate if a review is positive or not.

> Features:

1. count(positive lexicon words ∈ doc)
2. count(negative lexicon words ∈ doc)
3. { 1 – “no” found; 0 – Otherwise }
4. count(1st and 2nd pronouns ∈ doc)
5. { 1 – “!” found; 0 – Otherwise }
6. log(word count of doc)

Example: Sentiment Classification of movie reviews

It's hokey . There are virtually no surprises , and the writing is second-rate . So why was it so enjoyable ? For one thing , the cast is great . Another nice touch is the music . I was overcome with the urge to get off the couch and start dancing . It sucked me in , and it'll do the same to you .

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Example: Sentiment Classification of movie reviews

It's hokey . There are virtually no surprises , and the writing is second-rate . So why was it so enjoyable ? For one thing , the cast is great . Another nice touch is the music . I was overcome with the urge to get off the couch and start dancing . It sucked me in , and it'll do the same to you .

1.	count(positive lexicon words ∈ doc)	3
2.	count(negative lexicon words ∈ doc)	2
3.	{ 1 – "no" found; 0 – Otherwise }	1
4.	count(1st and 2nd pronouns ∈ doc)	3
5.	{ 1 – "!" found; 0 – Otherwise }	0
6.	log(word count of doc)	ln(66)=4.19

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Example: Sentiment Classification of movie reviews

Assuming we have a trained model with the following weights:

$$\beta_0 = 0.1, \qquad \beta = \begin{matrix} 2.5 \\ -5.0 \\ -1.2 \\ 0.5 \\ 2.0 \\ 0.7 \end{matrix}$$

1. 3
2. 2
3. 1
4. 3
5. 0
6. ln(66)=4.19

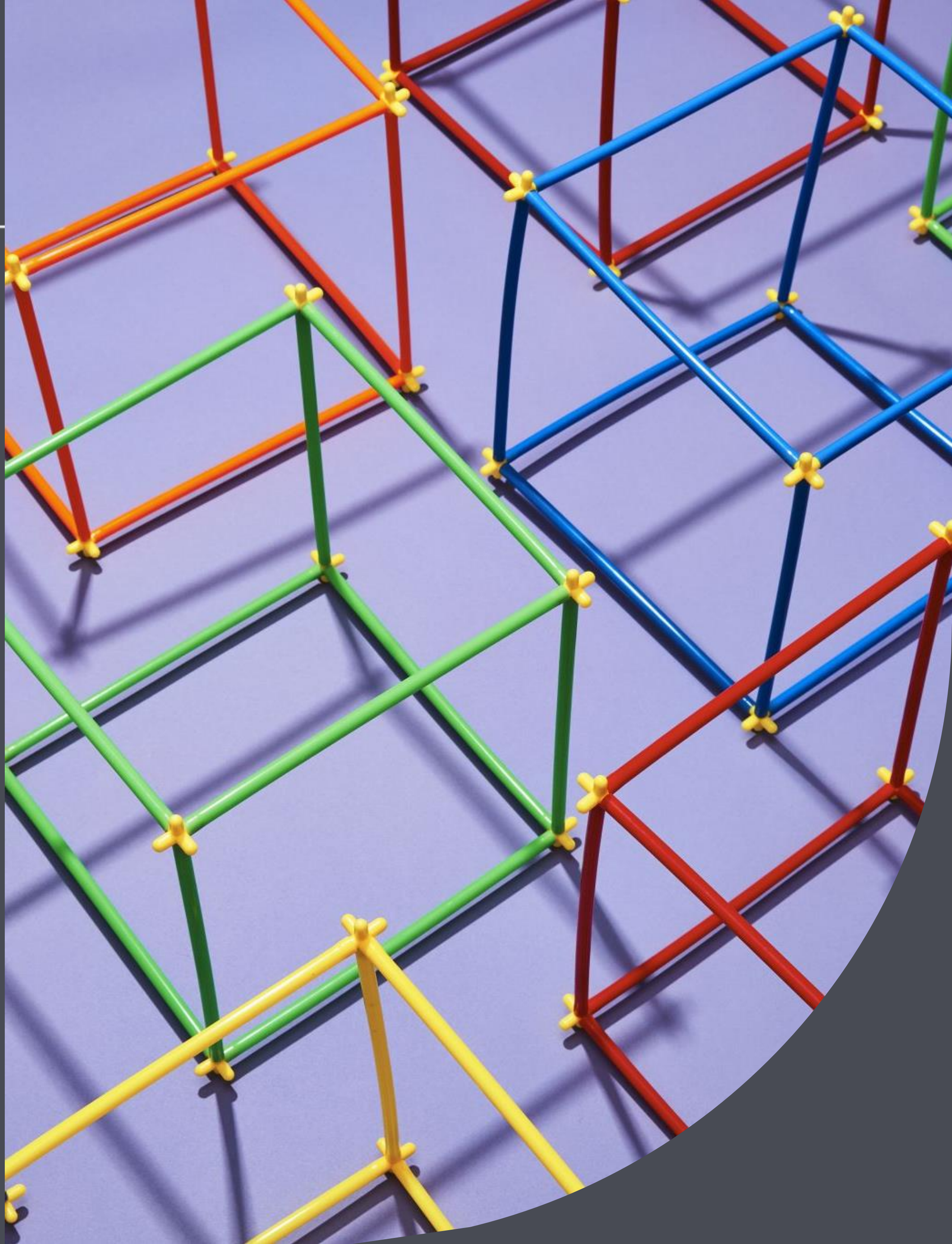
$$\sigma = \frac{1}{1 + e^{-(\beta_0 + \beta x)}}$$

$$P(y = 1|x) = \sigma(\beta \cdot x + \beta_0) = \sigma(.833) = 0.70$$

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Demo

> [A Neural Network Playground (tensorflow.org)]



What about multiclass?

> For *k* classes, we need *k* models
> y will be a 1-hot vector: [0 0 0 1 0 0 0 0 0]
> Multinomial Logistic Regression

Softmax instead of sigmoid:

$$softmax\ (z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)} = \frac{\exp(w_k \cdot x + b_k)}{\sum_{j=1}^{k} \exp(w_j \cdot x + b_j)}$$

What about multiclass?

Cross entropy loss:

$$-y \cdot log(p) + (1-y)\log(1-p)$$

Which generalizes to:

$$-\sum_{c=1}^{K} y_c \log(p_c)$$

Softmax instead of sigmoid:

$$softmax\ (z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)} = \frac{\exp(w_k \cdot x + b_k)}{\sum_{j=1}^{k} \exp(w_j \cdot x + b_j)}$$

Understand the math

> Demo

Linear Logistic Regression – Final Notes

- Logistic Regression:
  Linear regression wrapped with a non-linear function:
  - Sigmoid for binary classification
  - Softmax for multiclass classification

- Very fast to train
- Efficient in RAM consumption
- The basis of Neural Networks

Recommended Read & Watch for Deeper Dive

- Speech and Language Processing. Daniel Jurafsky & James H. Martin. Chapter 5 – Logistic Regression.pdf
- Logistic Regression -- Why sigmoid function? (sebastianraschka.com)
- (17) Softmax Function Explained In Depth with 3D Visuals - YouTube
- https://ml-cheatsheet.readthedocs.io/en/latest/logistic_regression.html
- Loss Functions — ML Glossary documentation (ml-cheatsheet.readthedocs.io)

Naïve Bayes

Classification with pure Bayesian statistics

Recap: Maximum Likelihood Estimation (MLE)

> Recall your MLE with Christian

Naïve Bayes – an Introduction

> Similar ideas to MLE

> A probabilistic **classifier**

  > (a **generative** one)

> Makes a naïve assumption about the features

> Assumes feature independence

> Uses the **Bayes Rule**:

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$$

Bayesian vs Frequentists

Bayesians:	Frequentists:
• Conditional Probability • Belief-based • A parameter is a random variable • Handle uncertainty with opinions • Goal: Update opinions	• Based on data • Parameters are static • Probability = How many times am I right? • Uncertainty requires data collection (null hypothesis) • Goal: decide which action to take

Bayesian vs Frequentists

Classification using Naïve Bayes

> Given a document, classify its author:

$$p(Mary\ Shelly \mid \text{"}The\ fallen\ angel\ becomes\ a\ malignant\ devil\text{"})$$

$$\hat{c} = \arg\max P(c|d) = \arg\max \frac{P(d|c)P(c)}{P(d)}$$

Since our document is the same for all classes, it can be simplified to:

$$\hat{c} = \arg\max P(c|d) = \arg\max P(d|c)P(c)$$

Classification using Naïve Bayes

> Given a document, classify its author:

$$p(Mary\ Shelly \mid \text{"}The\ fallen\ angel\ becomes\ a\ malignant\ devil\text{"})$$

$$\hat{c} = \arg\max P(c|d) = \arg\max \frac{P(d|c)P(c)}{P(d)}$$

Since our document is the same for all classes, it can be simplified to:

$$\hat{c} = \arg\max P(c|d) = \arg\max P(d|c)P(c)$$

Likelihood Prior

Classification using Naïve Bayes

> Given a document, classify its author:

$$p(Mary\ Shelly \mid \text{"The fallen angel becomes a malignant devil"})$$

$$\hat{c} = \arg\max P(c|d) = \arg\max P(sentence\ \uparrow)P(Mary\ Shelly)$$

Since our document is the same for all classes, it can be simplified to:

$$\hat{c} = \arg\max P(c|d) = \arg\max \underbrace{P(d|c)}_{\text{Likelihood}}\underbrace{P(c)}_{\text{Prior}}$$

Classification using Naïve Bayes

How are these probabilities obtained?

> Given a document, classify its author:

$$p(Mary\ Shelly \mid \text{"}The\ fallen\ angel\ becomes\ a\ malignant\ devil\text{"})$$

$$P(sentence \uparrow)P(Mary\ Shelly)$$
$$P(sentence \uparrow)P(Edgar\ Ellen\ Poe)$$
$$P(sentence \uparrow)P(HP\ Lovecraft)$$

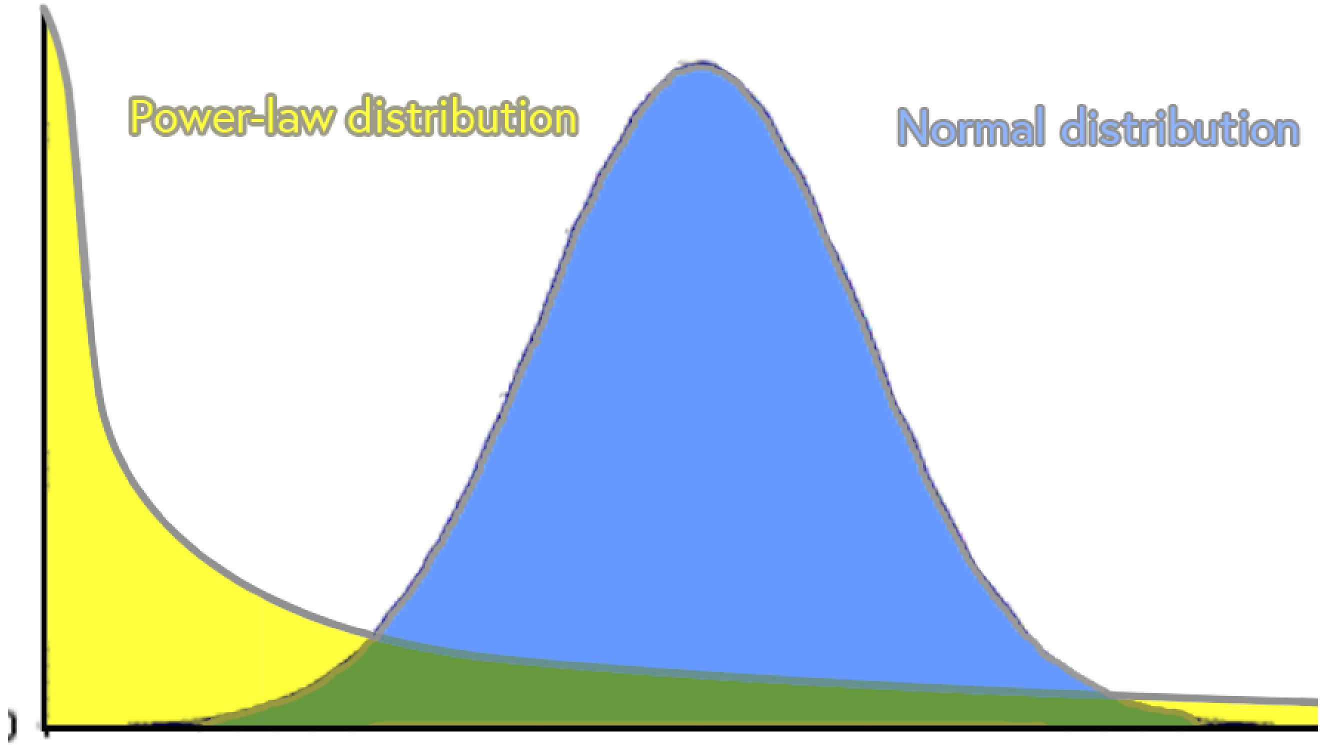
Since our document is the same for all classes, it can be simplified to:

$$\hat{c} = \arg\max P(c|d) = \arg\max P(d|c)P(c)$$

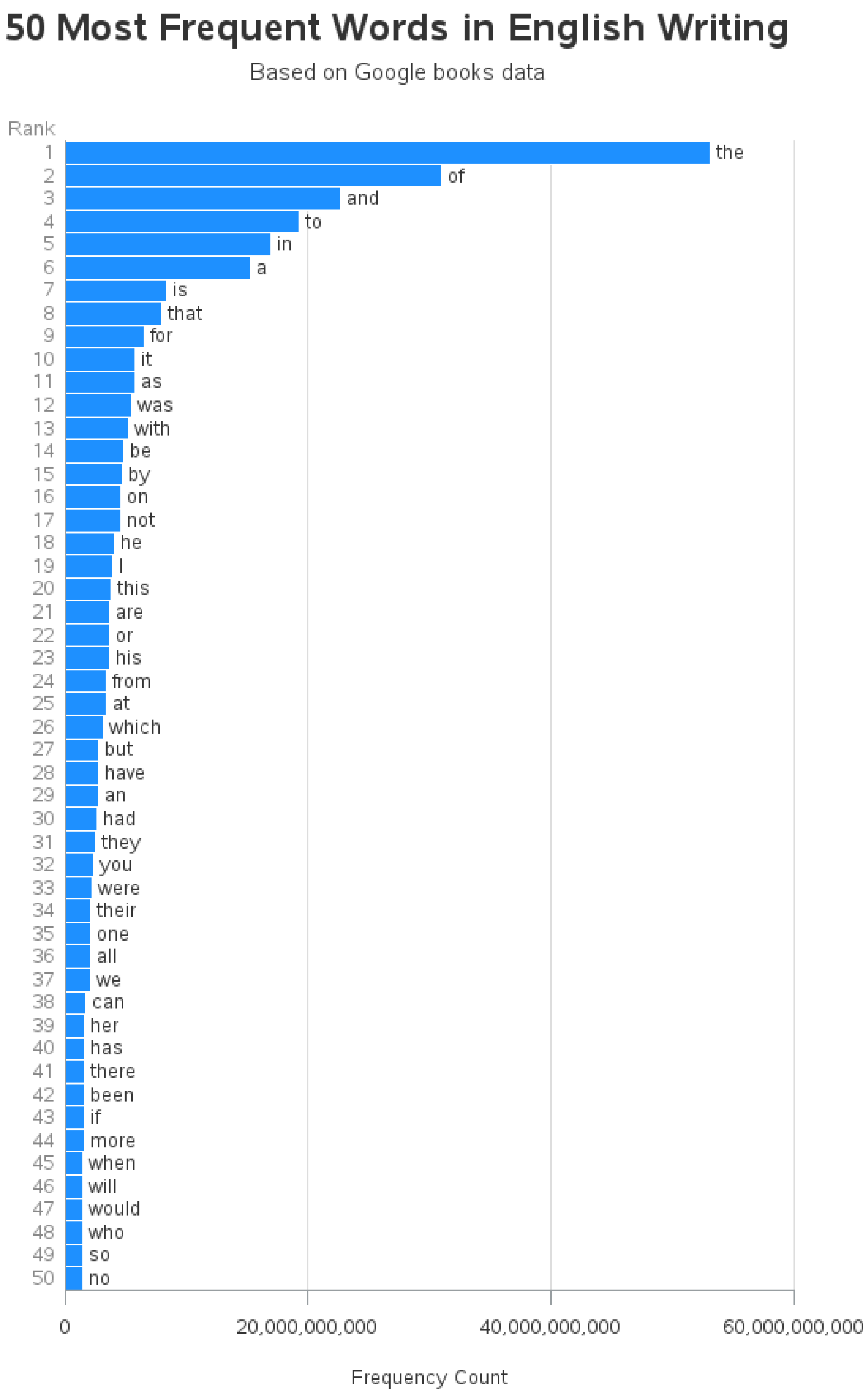
Likelihood Prior

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Language data properties



- Zipf Law
- Word frequencies follow a "power-law distribution":
  - **Long tail**
  - Most events rarely occur



Zipf Law

frequency of a word is inversely proportional to its rank in the frequency table

$$n(r) \propto \frac{1}{r^z} \qquad z \approx 1$$

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Zipf Law

In a 43M words text, there are:

- 316,710 unique words (types)
- 144,999 words occur only once
- 42,525 words occur 2 times
- 21,618 words occur 3 times
- 13,306 words occur 4 times
- 9,488 words occur 5 times
- 26,024 words appear >50 times

Zipf Law

No matter how large the training corpus is:

> It's likely to contain previously unseen word forms
> There will be many previously unseen word-*pairs*
> There will be even more previously unseen word-*triplets*
> There will be even more previously unseen *sentences*

Representing text as Features

"to be or not to be"

Bag of Words (BoW) - word counts:

{ "be": 2, "to": 2, "not": 1, "or": 1, "something": 0, "else": 0 }

(0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, … ,1, 0, 1, 0, …. ,0)

('a', 'the', 'there', 'to', …, 'be', 'if', 'an' … , 'or', 'as', 'not'…, 'zebra')

Representing text as Features

"to be or not to be"

Bag of Words (BoW) - word counts:

{ "be": 2, "to": 2, "not": 1, "or": 1, "something": 0, "else": 0 }

Reweighting:

TF/IDF or Pointwise Mutual Information – PMI

{ "be": 1.2, "to": 0.1, "not": 0.9, "or": 0.3 }

TF / IDF + PMI

{ "be": 2, "to": 2, "not": 1, "or": 1 }

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

$$PMI(a,b) = \log\left(\frac{P(a,b)}{P(\mathrm{a})\mathrm{P}(\mathrm{b})}\right)$$

Classification using Naïve Bayes

> Given a document, classify its author:

$$p(Mary\ Shelly \mid \text{"}The\ fallen\ angel\ becomes\ a\ malignant\ devil\text{"})$$

$$P(sentence\ \ \uparrow)P(Mary\ Shelly)$$

(Naively) Assuming feature independence:

$$P(sentence) =$$
$$P(\text{The}) \times P(\text{“}fallen\text{”}) \times$$
$$P(\text{“}angel\text{”}) \times P(\text{“}becomes\text{”}) \times P(\text{“}a\text{”}) \times$$
$$P(\text{“}malignant\text{”}) \times P(\text{“}devil\text{”})$$

Classification using Naïve Bayes

> Given a document, classify its author:

$$p(Mary\ Shelly\,|\,\text{"}The\ fallen\ angel\ becomes\ a\ malignant\ devil\text{"})$$

$$P(sentence\ \ \uparrow)P(Mary\ Shelly)$$

(Naively) Assuming feature independence:

$$P(sentence|Mary\ Shelly) =$$
$$P(\text{“}The\text{”}|Mary\ Shelly) \times P(\text{“}fallen\text{”}|Mary\ Shelly) \times$$
$$P(\text{“}angel\text{”}|Mary\ Shelly) \times P(\text{“}becomes\text{”}|Mary\ Shelly) \times P(\text{“}a\text{”}|Mary\ Shelly) \times$$
$$P(\text{“}malignant\text{”}|Mary\ Shelly) \times P(\text{“}devil\text{”}|Mary\ Shelly) = \prod_{w \in sentence} P(w|Mary\ Shelly)$$

What happens if a word doesn't appear in the text at all?

Classification using Naïve Bayes

> Given a document, classify its author:

$$p(Mary\ Shelly \mid \text{"}The\ fallen\ angel\ becomes\ a\ malignant\ devil\text{"})$$

$$P(sentence\ \uparrow)P(Mary\ Shelly)$$

(Naively) Assuming feature independence:

Obtaining an ***a-priori*** by counting word occurrences per class.

$$P(\text{“}devil\text{”} \mid Mary\ Shelly) = \frac{5}{1000}$$

"devil" occurrences

Total words in Mary Shelly documents

Classification using Naïve Bayes - Unseen Words

> Given a document, classify its author:

$$p(Mary\ Shelly\,|\,\text{"}The\ fallen\ angel\ becomes\ a\ malignant\ devil\text{"})$$

$$P(sentence\ \uparrow)P(Mary\ Shelly)$$

(Naively) Assuming feature independence:

$$\frac{0}{1000} = 0$$

$$P(sentence|Mary\ Shelly) =$$
$$P(\text{“}The\text{”}|Mary\ Shelly) \times P(\text{“}fallen\text{”}|Mary\ Shelly) \times$$
$$P(\text{“}angel\text{”}|Mary\ Shelly) \times P(\text{“}becomes\text{”}|Mary\ Shelly) \times P(\text{“}a\text{”}|Mary\ Shelly) \times$$
$$P(\text{“}malignant\text{”}|Mary\ Shelly) \times P(\text{“}devil\text{”}|Mary\ Shelly) \Rightarrow 0$$

Classification using Naïve Bayes - OoV Solution

> Given a document, classify its author:

$$p(Mary\ Shelly \mid \text{"}The\ fallen\ angel\ becomes\ a\ malignant\ devil\text{"})$$

$$P(sentence\ \uparrow)P(Mary\ Shelly)$$

(Naively) Assuming feature independence:

Obtaining an ***a-priori*** with *Laplace smoothing*:
Counting word occurrences + 1 per class,
divided by all class words + sentence-words.

$$P(\text{“}devil\text{”} \mid Mary\ Shelly) = \frac{5+1}{1000+7}$$

“devil” occurrences + 1

Total words in Mary Shelly documents + sentence’s tokens

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Classification using Naïve Bayes – Log Probability

> Working with log probability:

$$\hat{c} = \arg\max P(c|d) = \arg\max P(d|c)P(c)$$

For all classes and all words:

$$argmax_{c \in Classes} P(c) \prod_{w \in sentence} P(w_i|c)$$

To ease calculation and speed, we can take logs from both sides:

$$argmax_{c \in Classes} \log P(c) + \sum_{w \in sentence} \log P(w_i|c)$$

Math Recap: Log rules

$$\log_b(MN) = \log_b(M) + \log_b(N)$$

$$\log_b\left(\frac{M}{N}\right) = \log_b(M) - \log_b(N)$$

$$\log_b(M^p) = p\log_b(M)$$

Log rules: Justifying the logarithm properties (article) | Khan Academy

Naïve Bayes - Generative

> The bayes formula:

$$\hat{c} = \arg\max P(c|d) = \arg\max P(d|c)P(c)$$

Likelihood Prior

If the prior is known (category/class is given - $P(c) = 1$), we can generate words based on the Likelihood: $P(d|c)$

Reminder:
Discriminative → $P(\boldsymbol{c}lass|\boldsymbol{d}ocument)$ or

Generative → $P(\boldsymbol{d}ocument|\boldsymbol{c}lass)$

Naïve Bayes - Generative

On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes

Andrew Y. Ng
Computer Science Division
University of California, Berkeley
Berkeley, CA 94720

Michael I. Jordan
C.S. Div. & Dept. of Stat.
University of California, Berkeley
Berkeley, CA 94720

nips01-discriminativegenerative.pdf (stanford.edu)

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Naïve Bayes – Final Notes

> A linear & probabilistic classifier

> Implementation exists in NLTK & Scikit-Learn (SKL)

> On SKL one can choose the distribution. The best performance is normally the *Multinomial* Naive Bayes
[A Comparison of Event Models for Naive Bayes Text Classification: binomial.dvi (washington.edu)]

A Comparison of Event Models for Naive Bayes Text Classification

Andrew McCallum[‡†]
mccallum@justresearch.com
[‡]Just Research
4616 Henry Street
Pittsburgh, PA 15213

Kamal Nigam[†]
knigam@cs.cmu.edu

[†]School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Naïve Bayes – Final Notes

- Can act as a *discriminative* or *generative* model
- Still widely used in Linguistics for classification

- Good summary/deeper dive: [The Optimality of Naive Bayes]() – sections: Abstract & Naive Bayes and Augmented Naive Bayes (you can safely ignore the augmented Naïve Bayes)

The Optimality of Naive Bayes

Harry Zhang
Faculty of Computer Science
University of New Brunswick
Fredericton, New Brunswick, Canada E3B 5A3
email: hzhang@unb.ca

Dive deeper

- Bayes theorem, the geometry of changing beliefs – YouTube
- 1.9. Naive Bayes — scikit-learn 1.1.3 documentation
- 6. Learning to Classify Text (nltk.org)
- Dan Jourafsky’s book – Chapter 4: https://web.stanford.edu/~jurafsky/slp3/4.pdf