Neural Networks & Tensorflow.js

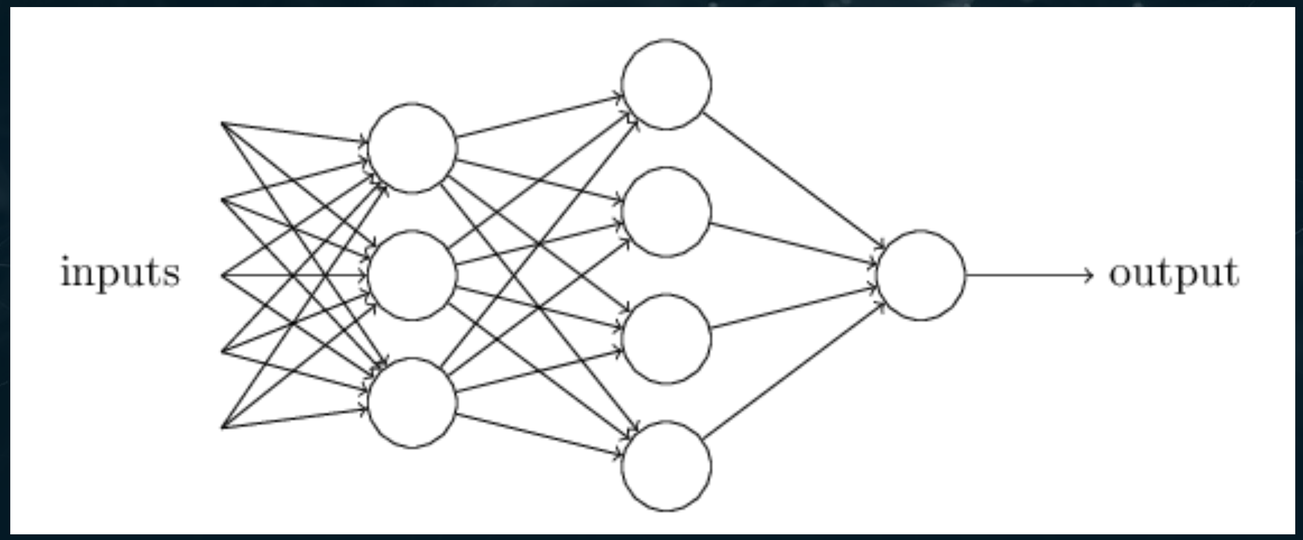
Magdalena Dembna

PART I

Short introduction to Neural Networks

Magdalena Dembna

Basic terminology - PERCEPTRONS

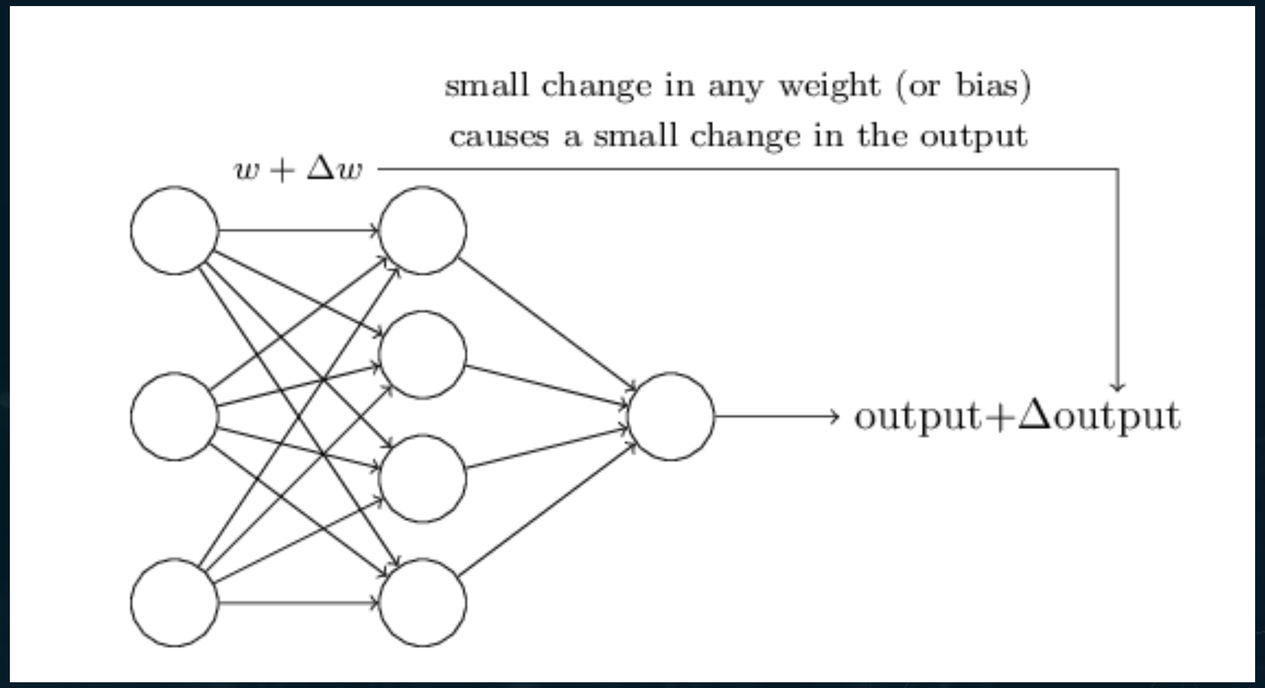


$$\text{output} = \begin{cases} 0 & \text{if } w \cdot x + b \leq 0 \\ 1 & \text{if } w \cdot x + b > 0 \end{cases}$$

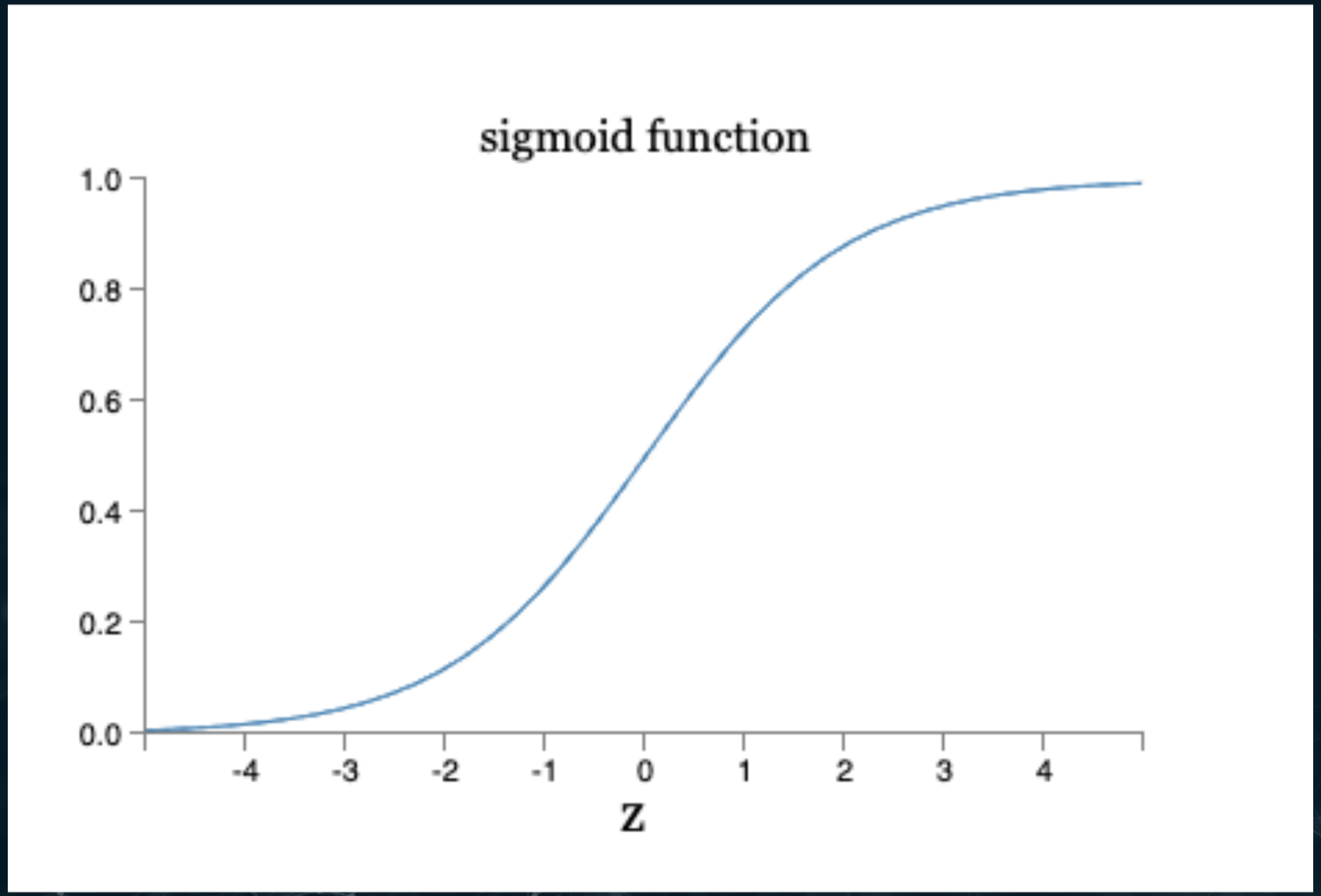
- A type of artificial neuron
- Takes several binary inputs and returns one binary output
- It can have several layers - each additional increased complexity of a model and decision making
- Small difference in weights/ biases can totally change output

w - WEIGHT (A real number expressing the importance of the respective input to the output)
b - BIAS (a measure how easy it is for the perceptron to fire)
x - INPUT (0 or 1)

Basic terminology - SIGMOID



- A type of artificial neuron
- Output and input are not binary but numbers between 0 and 1
- Change in output is a linear function of changes Δwj, Δbj in weights and biases - small adjustment in either one of those leads to a small change in output
- Sigmoid function (activation function) - function which takes a real number as an input and returns value between 1 and 0


sigmoid function
1.0
0.8
0.6
0.4
0.2
0.0
-4
-3
-2
-1
0
1
2
3
4
Z

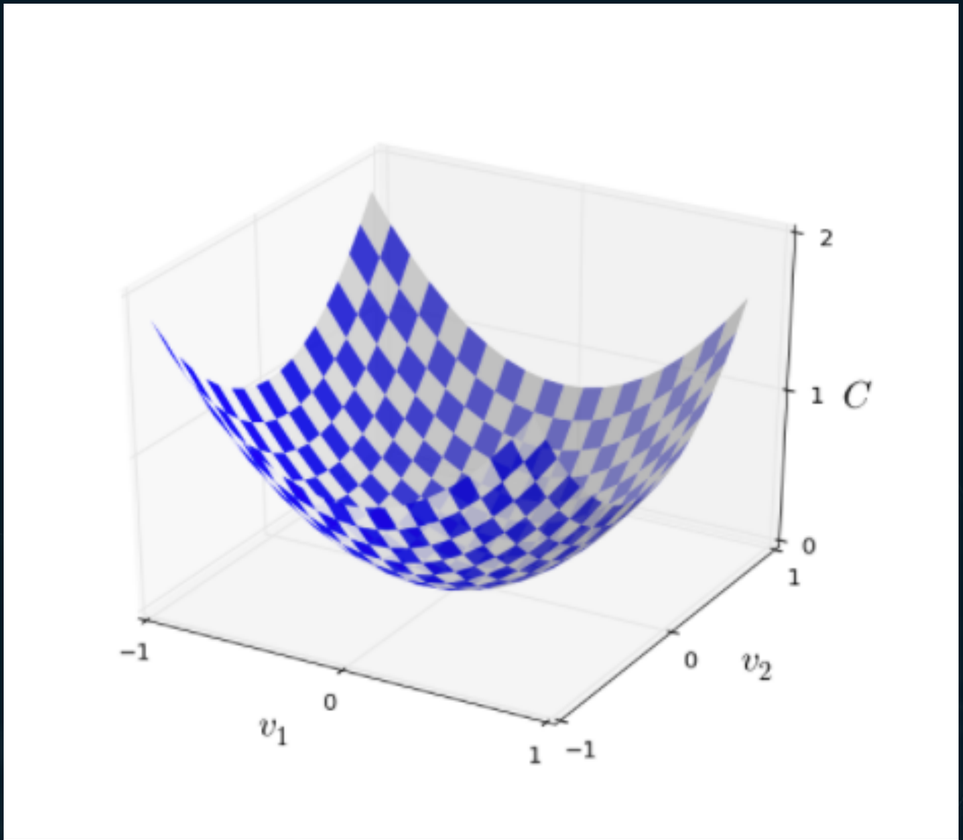
Basic terminology - COST FUNCTION

- **Measures the performance of a Machine Learning model for given data.** Cost Function quantifies the error between predicted values and expected values and presents it in the form of a single real number.

- Creating a successful neural networks requires finding set of weights and biases which minimise cost function

- Other names: loss function, objective function

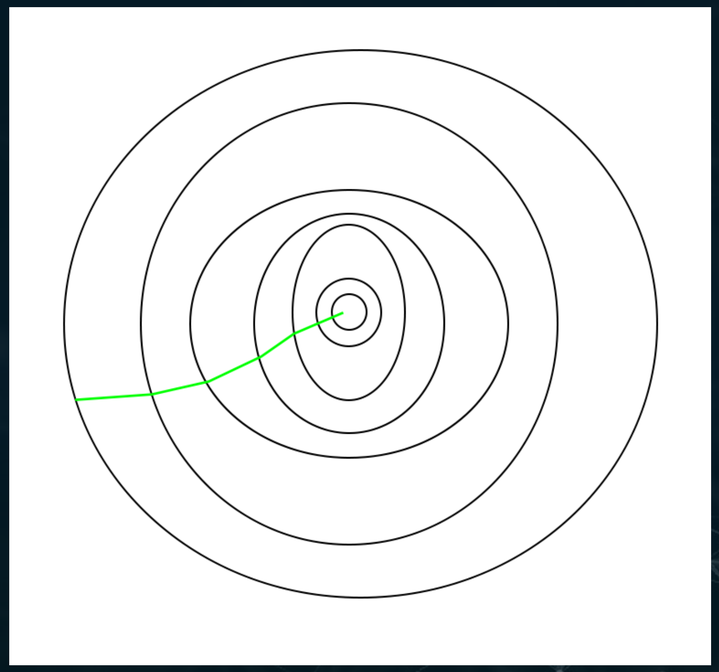
Basic terminology - GRADIENT DESCENT

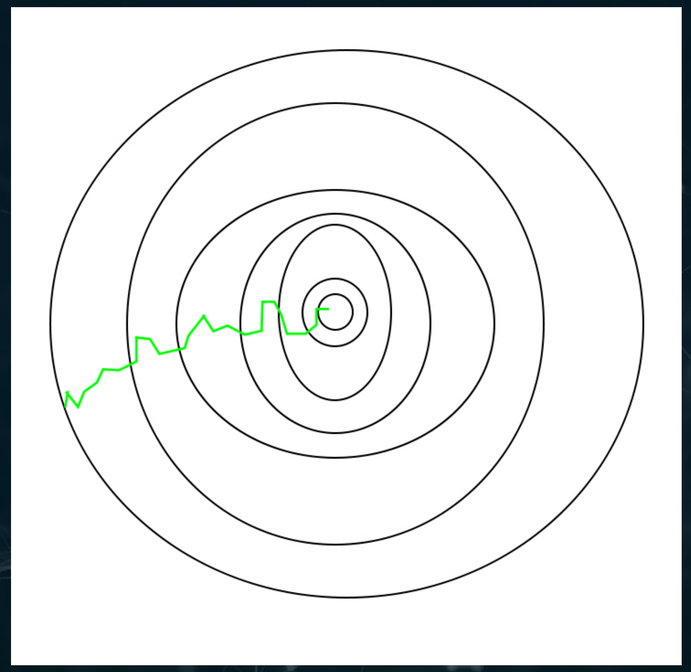


- We use gradient descent to find the weights and biases which minimise a value of the Cost function C(v)
- In Calculus we can compute derivates and find extrema for functions with few variables - in neural networks we need far more than few
- We’re trying to find a local minimum by choosing random point and calculating “downhill” direction
- It’s quite costly - we need to compute gradient for each training input and it can take a long time

Basic terminology - STOCHASTIC GRADIENT DESCENT

- Optimisation for large data sets, often used in ML algorithms
- We split training inputs into **mini batches** (small sample of randomply chosen inputs) and calculate GD for them





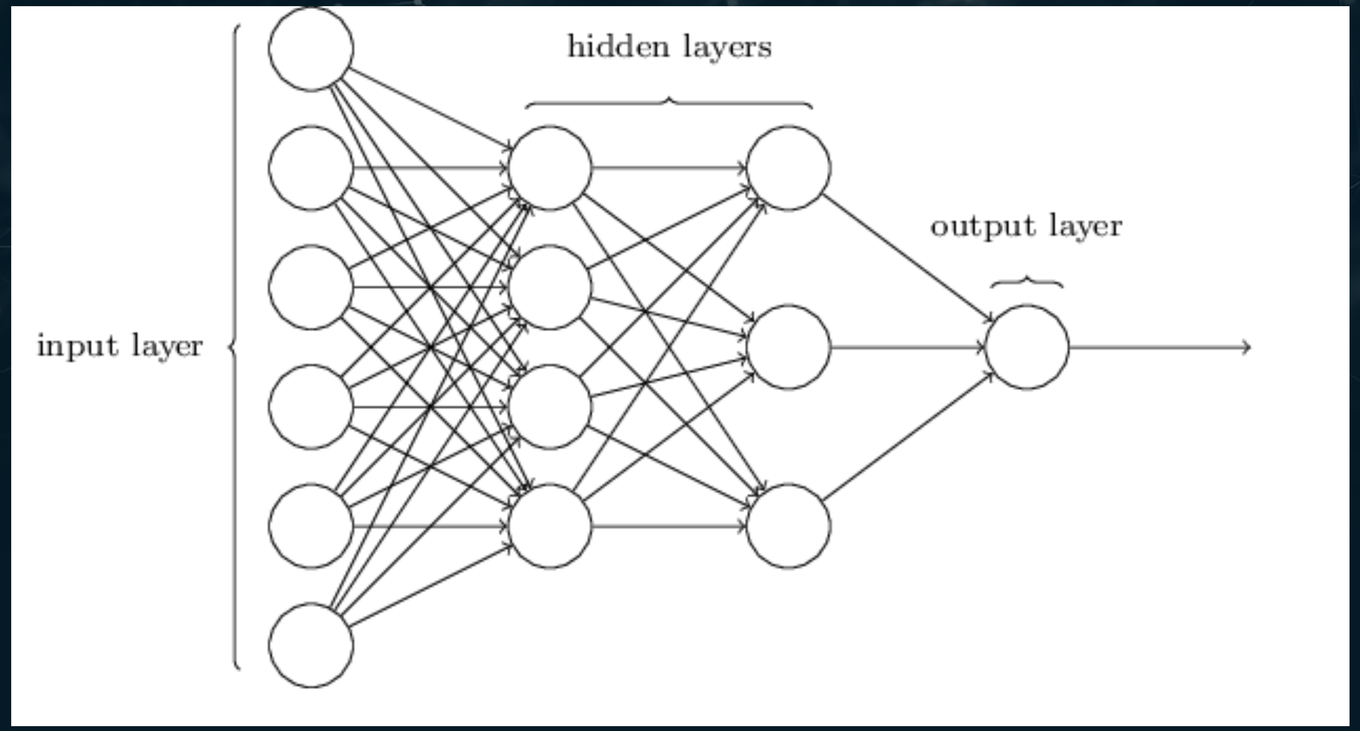
Basic terminology - BACKPROPAGATION

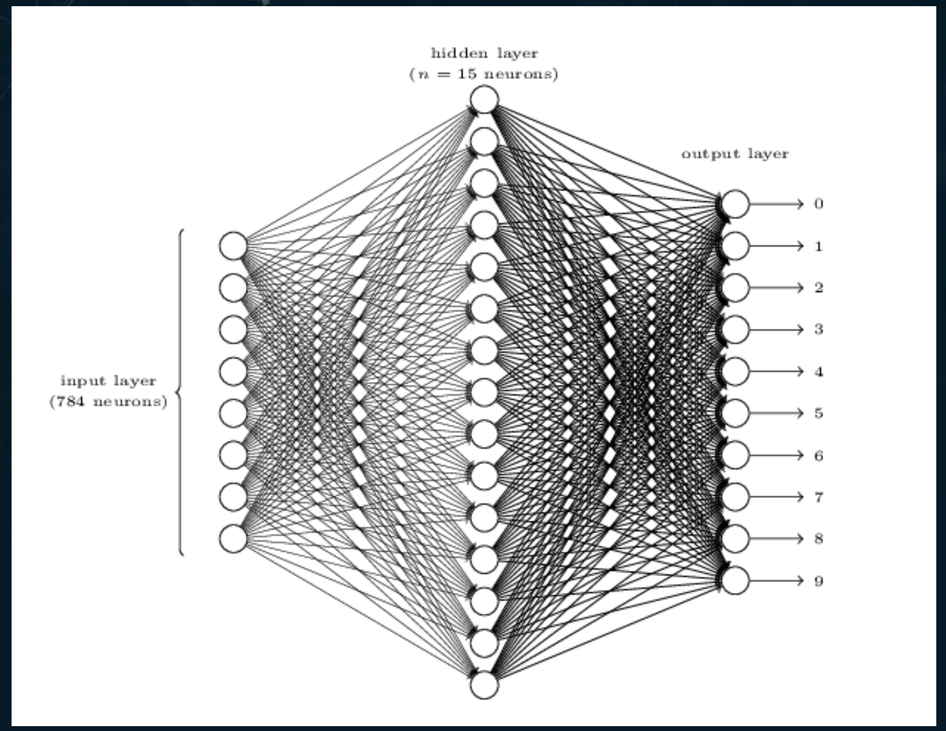
- **An algorithm used to effectively train a neural network through a method called chain rule.** In simple terms, after each forward pass through a network, backpropagation performs a backward pass while adjusting the model's parameters (weights and biases).

- B**ackpropagation aims to minimize the cost function by adjusting network's weights and biases.** The level of adjustment is determined by the gradients of the cost function with respect to those parameters.

- Read more: https://towardsdatascience.com/understanding-backpropagation-algorithm-7bb3aa2f95fd

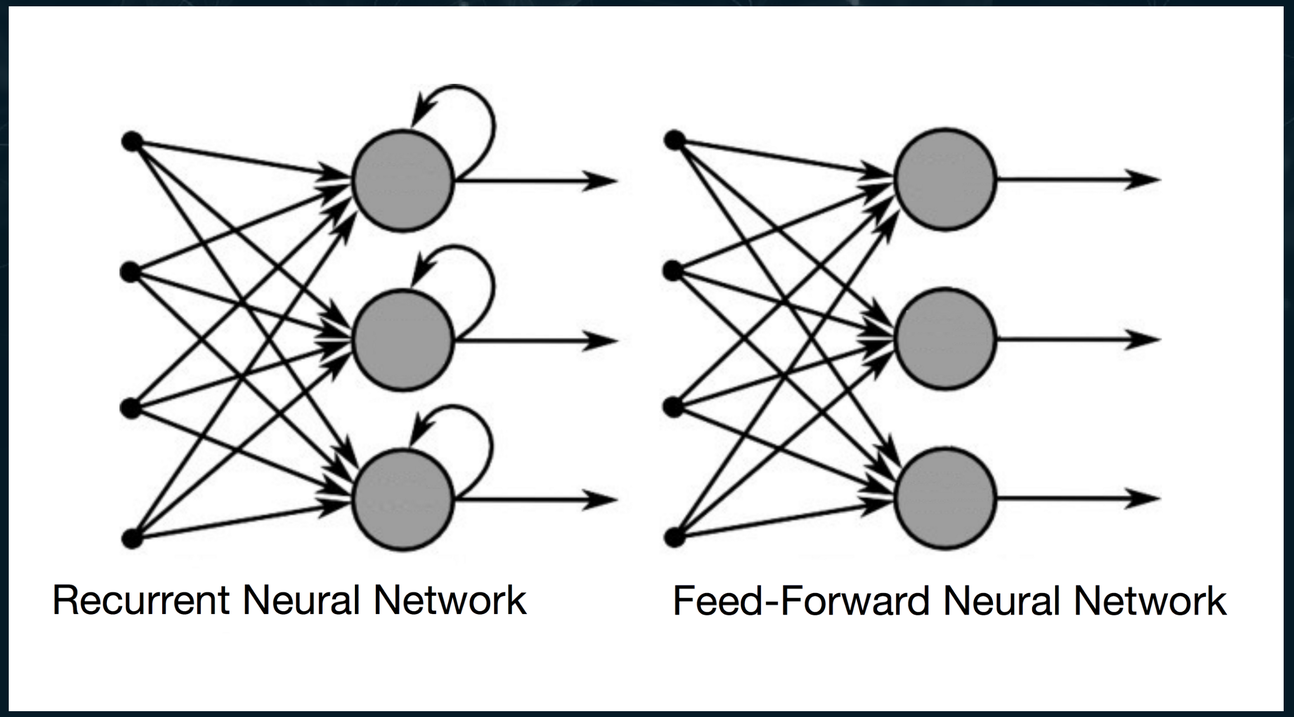
ARCHITECTURE





- **Input layer** - consists of input neurons (perceptrons or sigmoids)
- **Output layer** - single output neuron (in this case) or several outputs with different activation
- **Hidden layers** - everything in between

BASIC TYPES OF NEURAL NETWORKS



- **feed-forward neural networks** - output from previous layer is used as an input in the next one - no loops in the network

- **recurrent neural networks** - neurons fire in limited duration of time - firing stimulates other neurons with some delay

PART II

Tensorflow.js

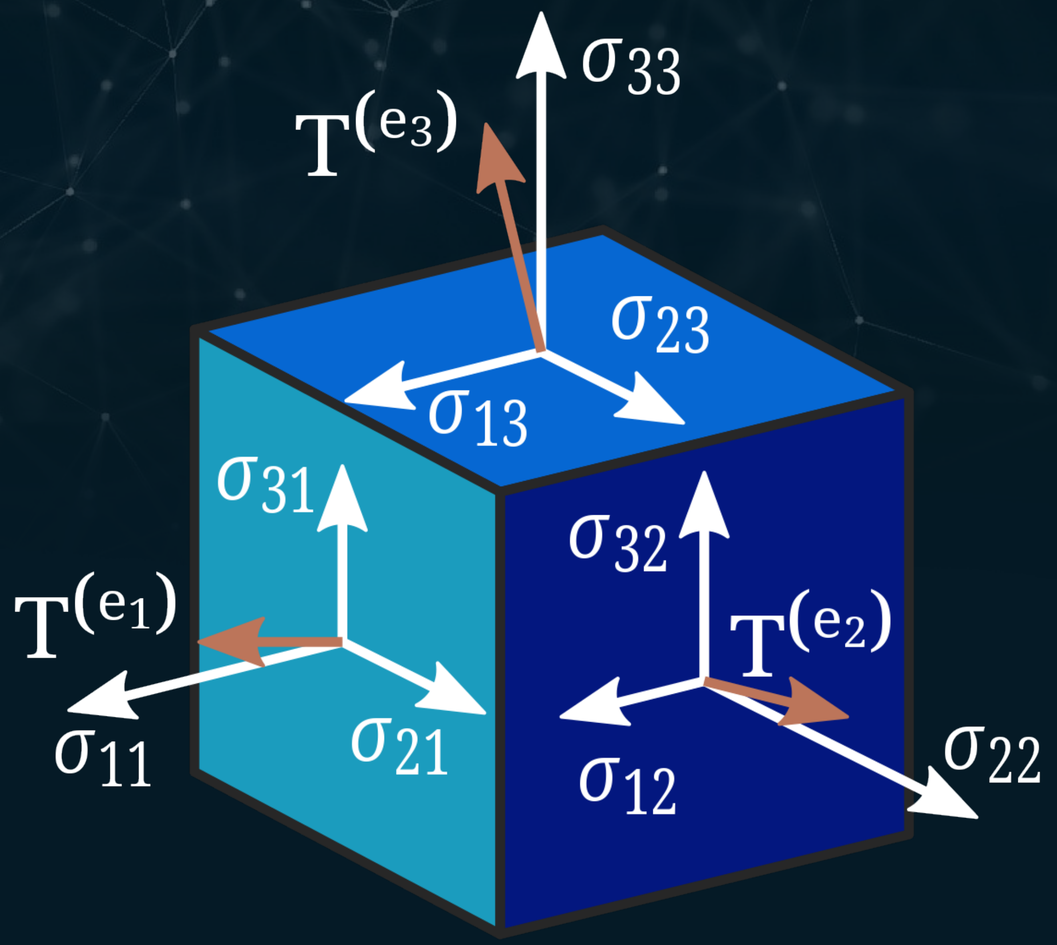
Magdalena Dembna

What’s a tensor?

Tensors are the core datastructure of TensorFlow.js.
They are a generalization of vectors and matrices to potentially higher dimensions.
They can be easily understood as a multidimensional arrays.

[1 2] <- matrix

[[1 2] [3 4]] <- tensor



Neural Networks in a browser

- Running existing models
- Retraining existing models
- **Build and train models directly in Javascript ***

* In-browser models shouldn't deal with datasets bigger than 30mb

Training models in Javascript - example

https://codelabs.developers.google.com/codelabs/tfjs-training-regression/index.html#0

Exercise: Given "Horsepower" for a car, the model will learn to predict "Miles per Gallon" (MPG)

To do this we will:

- Load the data and prepare it for training
- Define the architecture of the model.
- Train the model and monitor its performance as it trains
- Evaluate the trained model by making some predictions

Training models in Javascript - example

https://codelabs.developers.google.com/codelabs/tfjs-training-regression/index.html#0

Exercise: Given "Horsepower" for a car, the model will learn to predict "Miles per Gallon" (MPG)

To do this we will:

- Load the data and prepare it for training
- Define the architecture of the model.
- Train the model and monitor its performance as it trains
- Evaluate the trained model by making some predictions

Add Tensorflow.js to your project

https://www.npmjs.com/package/@tensorflow/tfjs

```
npm i @tensorflow/tfjs

import * as tf from "@tensorflow/tfjs";
```

Load data

To train neural network we need a dataset.

In this example we use data from Tensorflow.js tutorial: https://storage.googleapis.com/tfjs-tutorials/carsData.json - it is a simple array of objects with two fields - `horsepower` and `mpg` (miles per gallon).

The most commonly used dataset for learning ML is MNIST - http://yann.lecun.com/exdb/mnist/ - it consists images of hand-written digits with corresponding labels.

Define the architecture

- **Sequential modal** - a model appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor

```
const model = tf.sequential();
```

- **Single input layer** with one hidden layer and 1 number as an input (inputShape = 1). One weight will be assigned to each input (units = 1)

```
model.add(tf.layers.dense({inputShape: [1], units: 1, useBias: true}));
```

- One number as an **output** (units = 1)

```
model.add(tf.layers.dense({units: 1, useBias: true}));
```

Prepare data for training

1. Convert to tensors

We make two arrays, one for our input examples (the horsepower entries), and another for the true output values (which are known as labels in machine learning).

2. Shuffle data

Shuffling is important because typically during training the dataset is broken up into smaller subsets, called batches, that the model is trained on. Shuffling helps each batch have a variety of data from across the data distribution.

3. Normalize the data to the range 0 - 1 using min-max scaling

Prepare for training

Choose following properties:

- **Optimiser** (Adam, SGD, or other: https://js.tensorflow.org/api/latest/#Training-Optimizers)
- **Loss/Cost function** (f.e. Mean Squared Error *https://developers.google.com/machine-learning/glossary/#MSE*)
- **Batch size** - a size of the data subsets that the model will see on each iteration of training
- **Epochs** - he number of times the model is going to look at the entire dataset that you provide it.

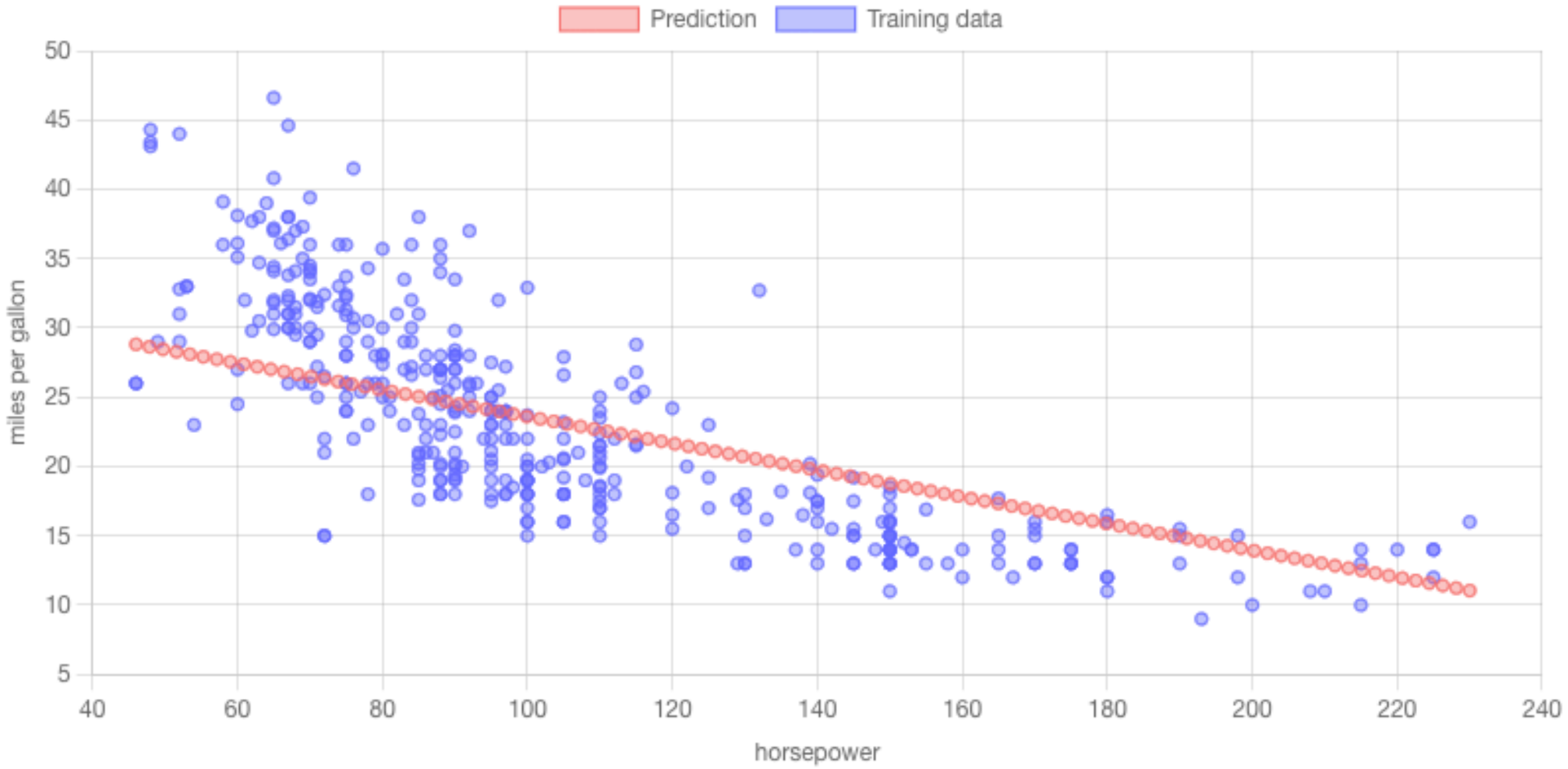
Start the training loop

Let’s call `model.fit()` and wait fot Tensorflow to do its magic…

Explaining what happens under the hood is far beyond the scope of my presentation - If you want to learn more there are useful links on the last slide.

Predictions

- Generate new data examples and let the model predict values for each of them
- Values are in range between 0 and 1 so we need to invert normalising operations
- Plot predicted points on a graph with training dataset to visualise effects


Prediction
Training data
50
45
40
35
30
25
20
15
10
5
miles per gallon
40
60
80
100
120
140
160
180
200
220
240
horsepower

THE END

Sources

- *https://www.geeksforgeeks.org/ml-stochastic-gradient-descent-sgd/*
- *https://towardsdatascience.com/understanding-backpropagation-algorithm-7bb3aa2f95fd*
- *http://neuralnetworksanddeeplearning.com/chap1.html*
- *http://yann.lecun.com/exdb/mnist/*
- *https://www.tensorflow.org/js*
- *https://codelabs.developers.google.com/codelabs/tfjs-training-regression/index.html#0*

Learn more

- *3 blue 1 brown YouTube channel: https://www.youtube.com/watch?v=IHZwWFHWa-w*
- *EDX courses*