Exam

We have >750 registered
students for the exam
on 27th of March 2024.

We keep you updated about
the exam modality
(single choice + X)

Evaluation

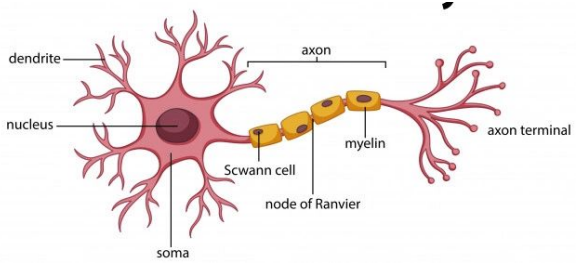
Please take part in the evaluation, it means a lot to us :-)

Data Science Survival Skills

Artificial Intelligence, Machine Learning, Deep Learning

Part II:
Deep Learning

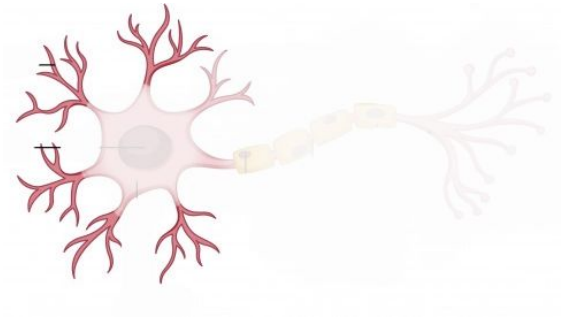
Perceptrons

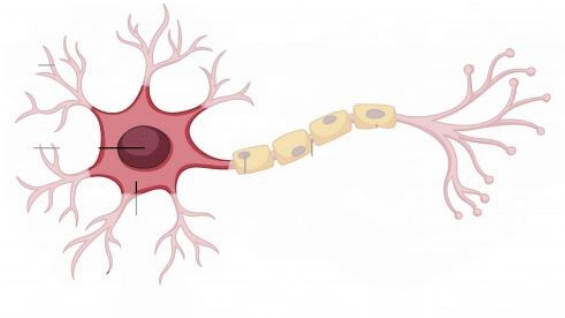


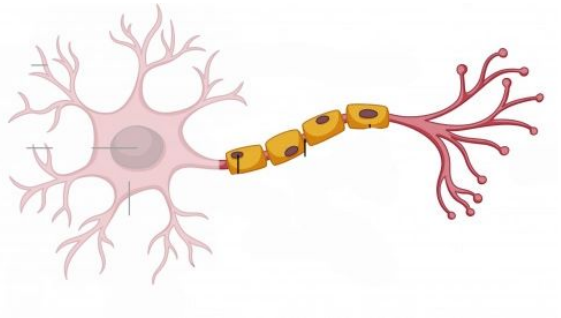
INPUT

COMPUTATION

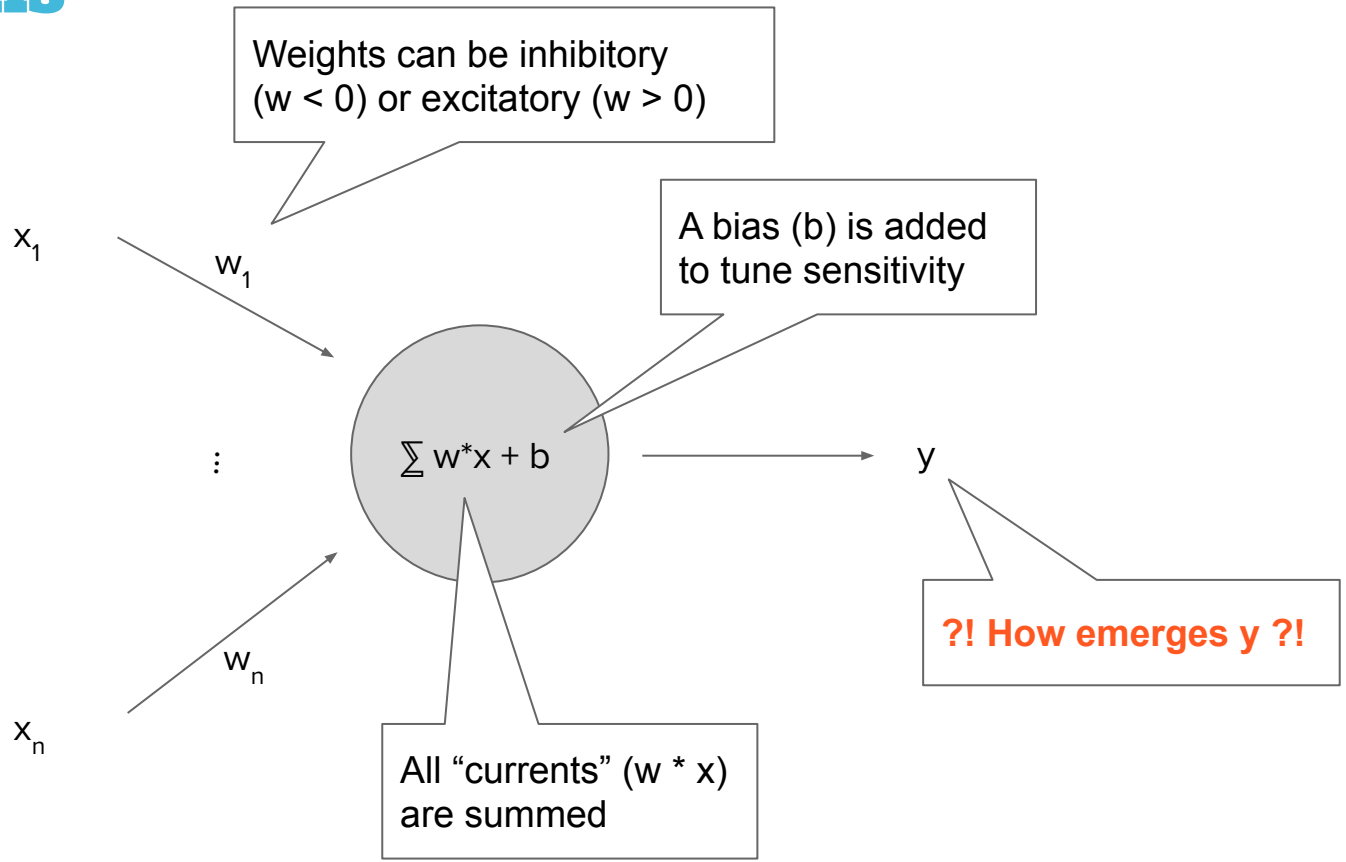
OUTPUT







Perceptrons



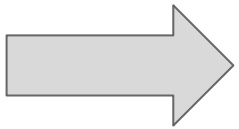
Perceptrons

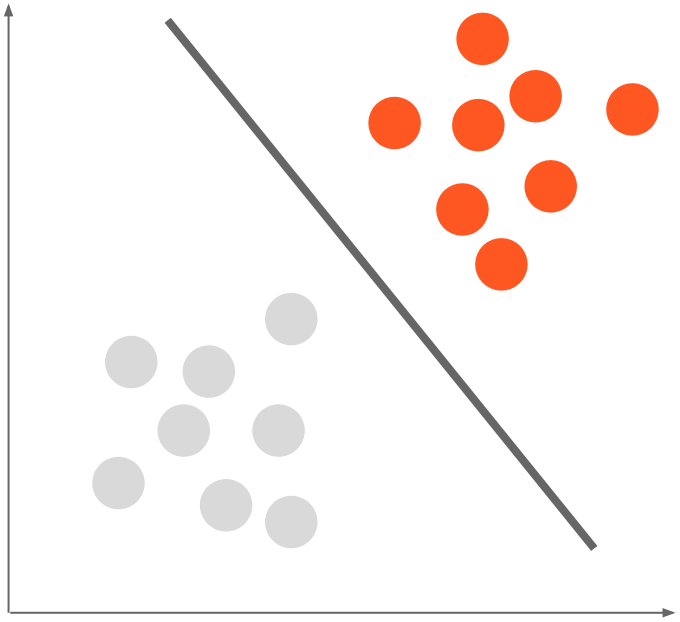
Perceptrons are **LINEAR CLASSIFIER**!

Threshold
step function:

$$\sum_{i=1}^{m} w_i x_i$$

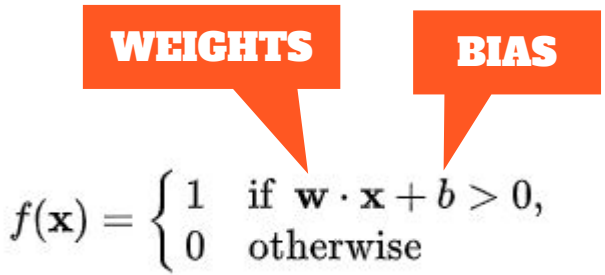
$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b > 0, \\ 0 & \text{otherwise} \end{cases}$$



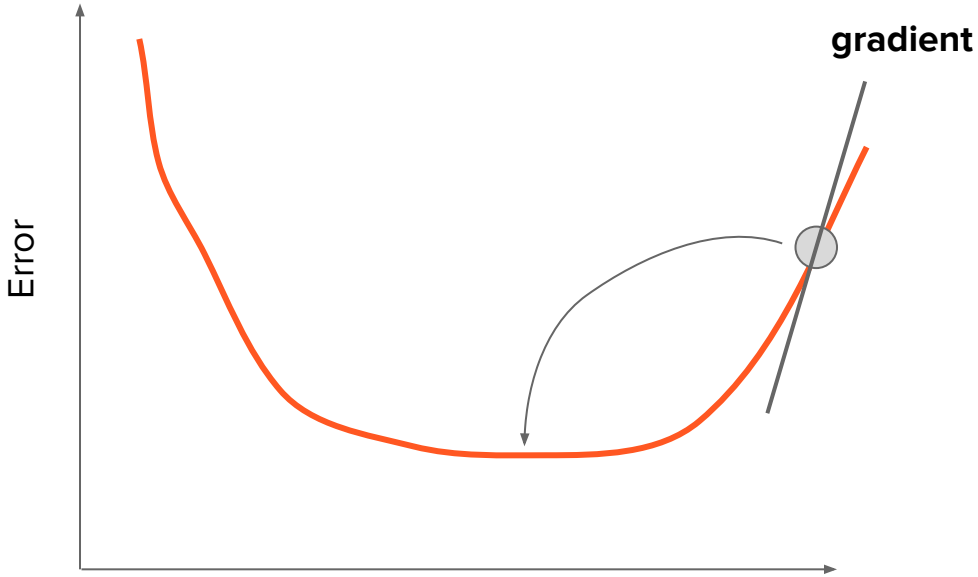


Teaching and Learning

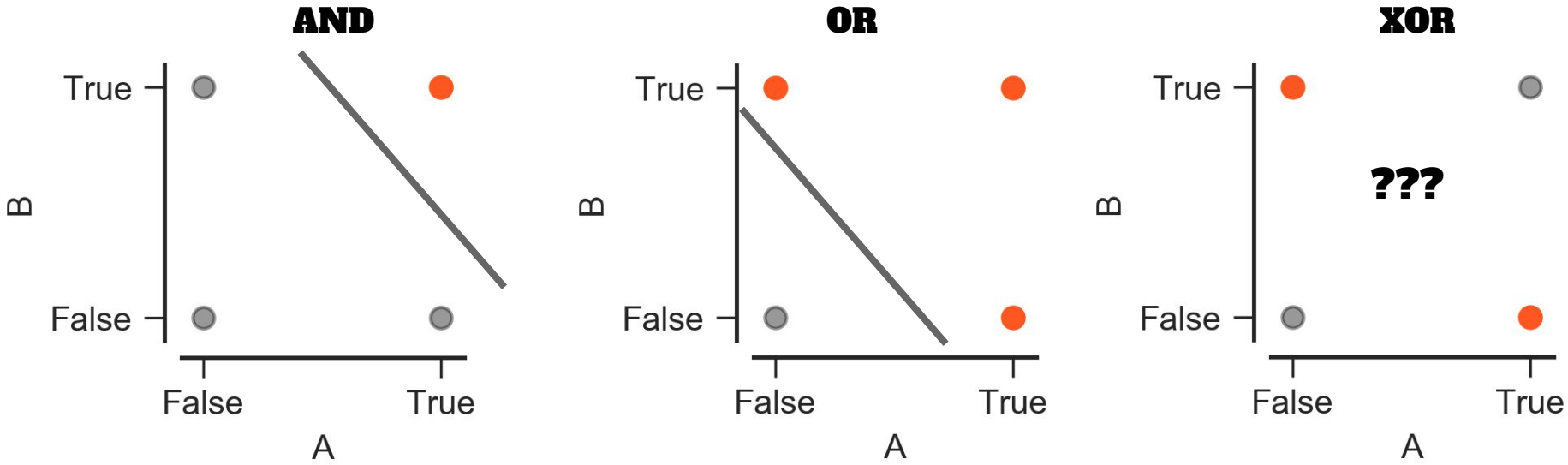
What are the parameters one could tune?



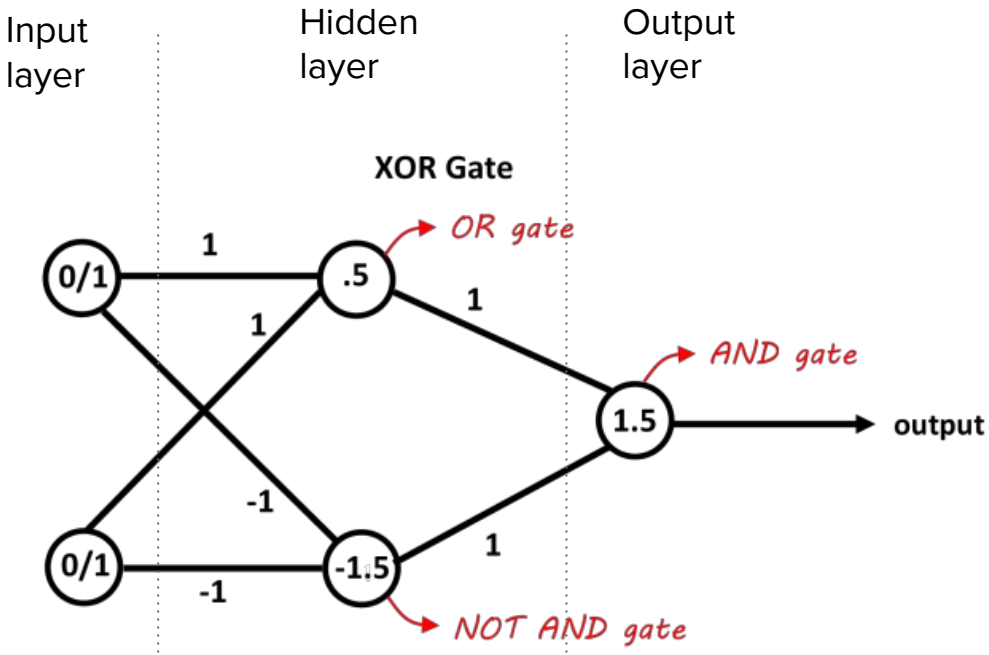
Learning weights with gradient descent

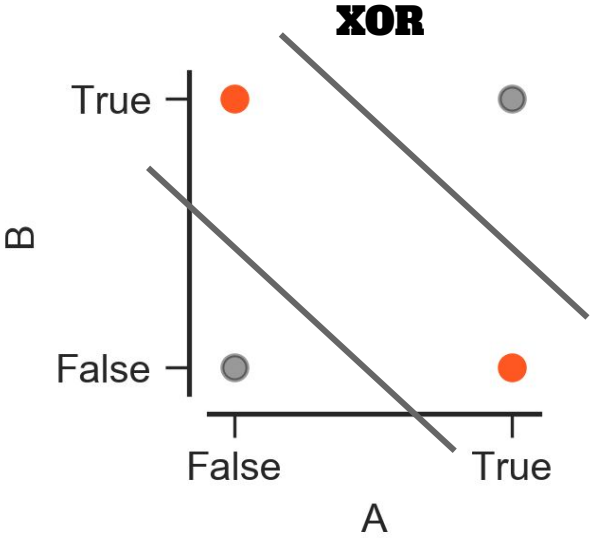


Why perceptrons were abandoned...

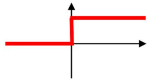
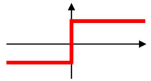
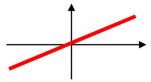
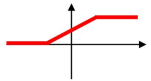
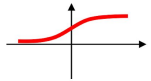
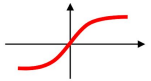
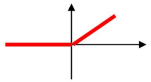
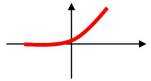


But there is a solution: The multilayer perceptron!





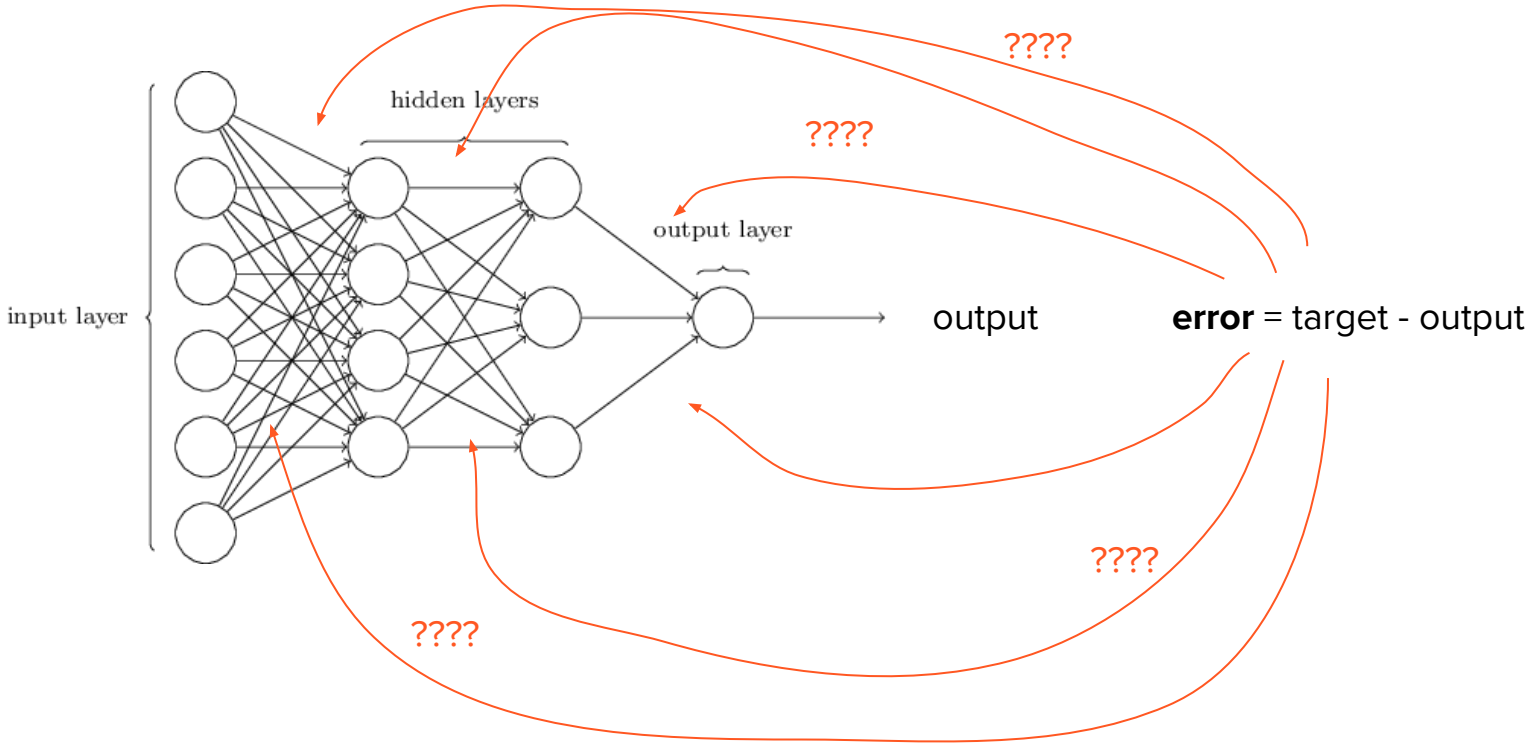
Activation functions

Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer Neural Networks	
Rectifier, ReLU (Rectified Linear Unit)	$\phi(z) = max(0, z)$	Multi-layer Neural Networks	
Rectifier, softplus	$\phi(z) = \ln(1 + e^z)$	Multi-layer Neural Networks	

Copyright © Sebastian Raschka 2016 (http://sebastianraschka.com)

Assigning the error

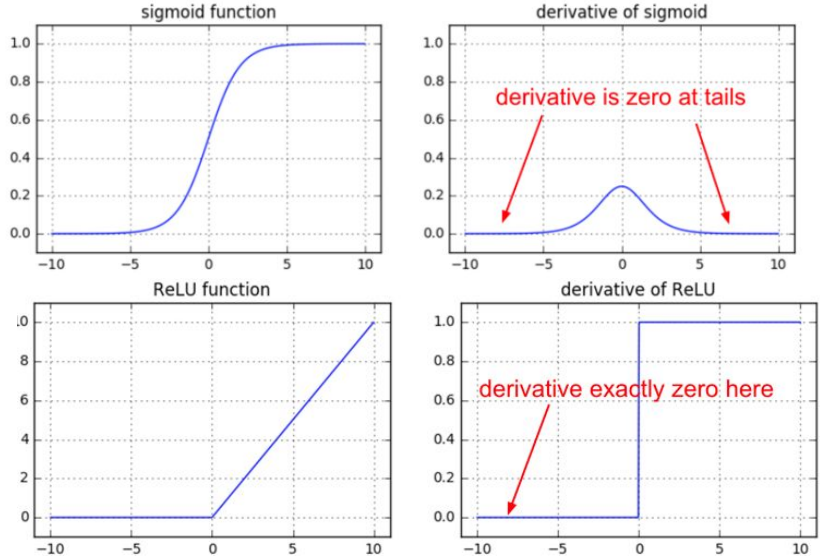
➜ Something called backpropagation.
It is done automatically, but you should KNOW WHAT IT IS DOING!



Yes you should understand backprop

Andrej Karpathy Dec 19, 2016 · 7 min read

17K 46



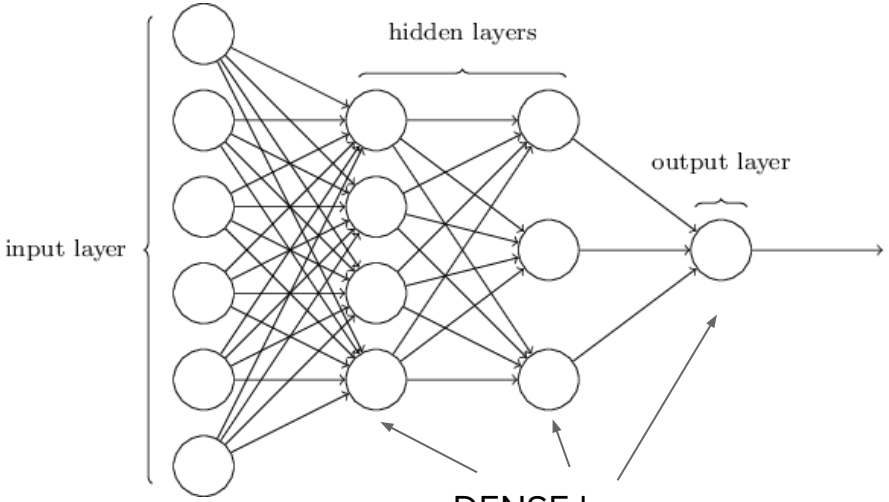
In conclusion

Backpropagation is a leaky abstraction; it is a credit assignment scheme with non-trivial consequences. If you try to ignore how it works under the hood because "TensorFlow automagically makes my networks learn", you will not be ready to wrestle with the dangers it presents, and you will be much less effective at building and debugging neural networks.

https://medium.com/@karpathy/yes-you-should-understand-backprop-e2f06eab496b

TensorFlow/Keras

TensorFlow



e.g. CAT/DOG

DENSE layers

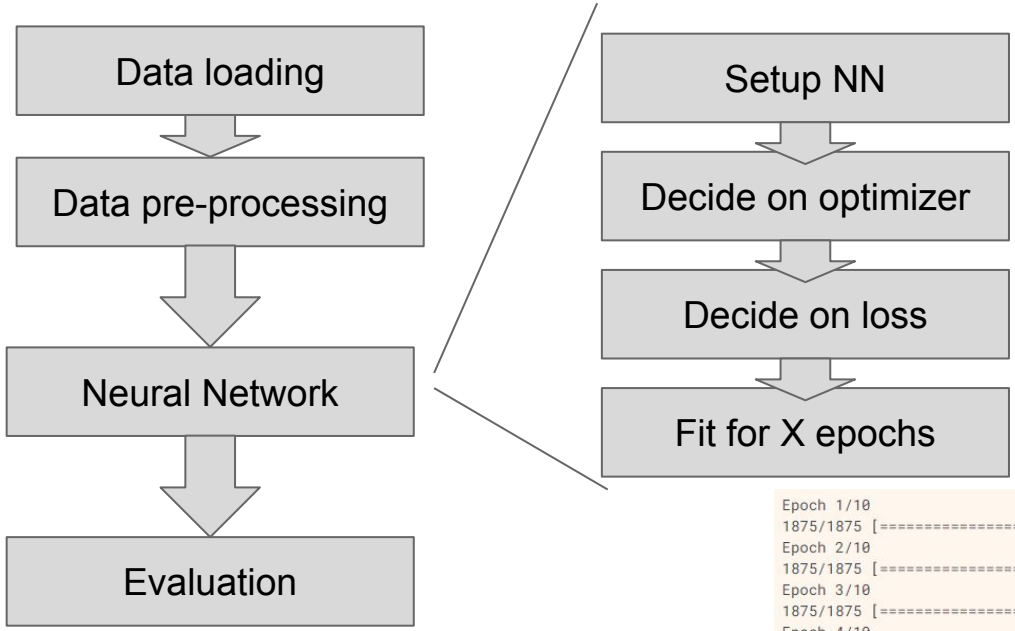
1	1	0
4	2	1
0	2	1

1
1
0
4
2
1
0
2
1

INPUT layer

FLATTEN layer

How to create a NN (in Keras)




```
model = tf.keras.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(10)
])
```

```
model.compile(optimizer='adam',
              loss=SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

```
model.fit(train_images, train_labels, epochs=10)
```

```
Epoch 1/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.4917 - accuracy: 0.8277
Epoch 2/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.3702 - accuracy: 0.8674
Epoch 3/10
1875/1875 [==============================] - 3s 2ms/step - loss: 0.3328 - accuracy: 0.8793
Epoch 4/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.3106 - accuracy: 0.8859
```

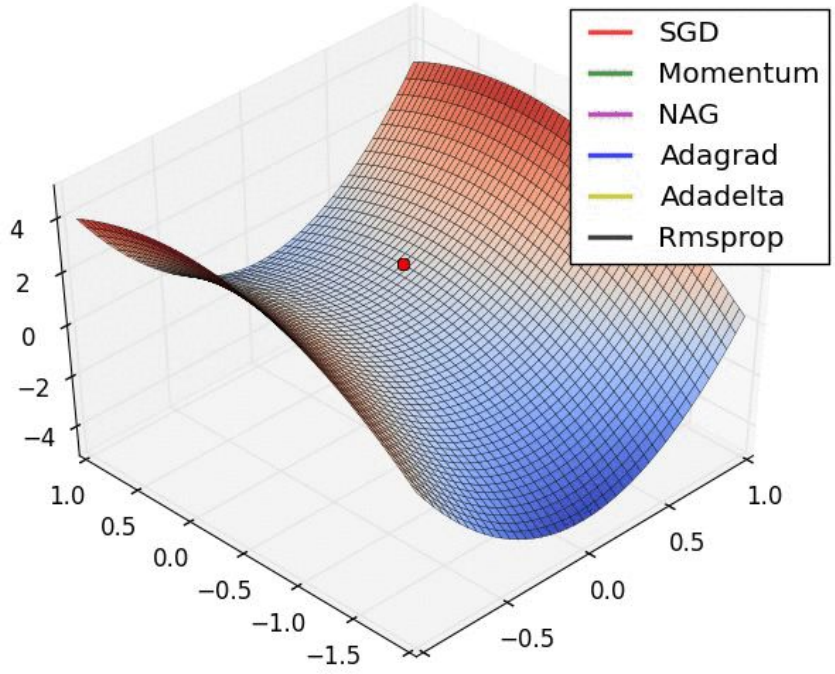
```
313/313 - 1s - loss: 0.3637 - accuracy: 0.8693

Test accuracy: 0.8693000078201294
```

Optimizers

Strategies to optimize the network and perform gradient descent:

- Stochastic gradient descent (SGD)
- SGD + momentum
- RMSprop
- Adadelta
- Adam
- ...

Try SGD(momentum=.9) and Adam



The loss function

Important:
your objective function should represent what you would like to optimize!!

Regression Losses

Mean Square Error/Quadratic Loss/L2 Loss

Mathematical formulation :-

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

Mean Squared Error

Cross Entropy Loss/Negative Log Likelihood

This is the most common setting for classification problems. Cross-entropy loss increases as the predicted probability diverges from the actual label.

Mathematical formulation :-

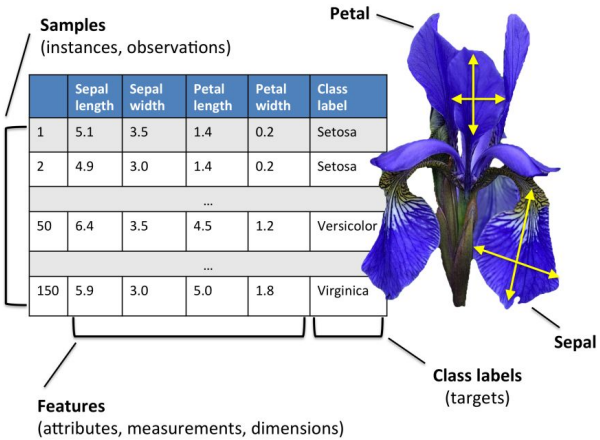
$$CrossEntropyLoss = -(y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i))$$

Cross entropy loss

Code for an MLP

```
model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10, activation='softmax')
]) # Multilayer perceptron
```

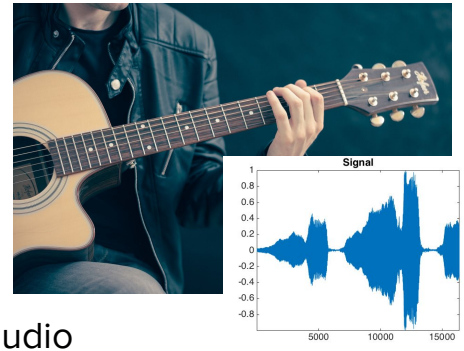
What kind of data do you have…?



	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
			...		
50	6.4	3.5	4.5	1.2	Versicolor
			...		
150	5.9	3.0	5.0	1.8	Virginica

Structured data!

images



audio

text

esf evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ MLÁDEŽE A TĚLOVÝCHOVY
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Johann Wolfgang Goethe: *Erlkönig* -

Wer reitet so spät durch Nacht und Wind?
Es ist der Vater mit seinem Kind;
Er hat den Knaben wohl in dem Arm,
Er fasst ihn sicher, er hält ihn warm.

„Mein Sohn, was birgst du so bang dein Gesicht?" –
„Siehst, Vater, du den Erlkönig nicht?
Den Erlenkönig mit Kron´ und Schweif?" –
„Mein Sohn, es ist ein Nebelstreif." –

„Du, liebes Kind, komm, geh mit mir!
Gar schöne Spiele spiel´ ich mit dir;
Manch bunte Blumen sind an dem Strand,
Meine Mutter hat manch gülden Gewand."

„Mein Vater, mein Vater, und hörest du nicht,
Was Erlenkönig mir leise verspricht?" –
„Sei ruhig, bleibe ruhig, mein Kind;
In dürren Blättern säuselt der Wind." –

„Willst, feiner Knabe, du mit mir gehen?
Meine Töchter sollen dich warten schön;
Meine Töchter führen den nächtlichen Reihn
Und wiegen und tanzen und singen dich ein."

„Mein Vater, mein Vater, und siehst du nicht dort
Erlkönigs Töchter am düstern Ort?" –
„Mein Sohn, mein Sohn, ich seh es genau:
Es scheinen die alten Weiden so grau." –

„Ich liebe dich, mich reizt deine schöne Gestalt;
und bist du nicht willig, so brauch ich Gewalt." –
„Mein Vater, mein Vater, jetzt fasst er mich an!
Erlkönig hat mir ein Leids getan!" –

Dem Vater grauset´s, er reitet geschwind,
Er hält in den Armen das ächzende Kind,
Erreicht den Hof mit Mühe und Not;
In seinen Armen das Kind war tot.

Images

Imagine a noisy pixel

	12	11	11
v	13	255	10
	14	15	10

u

$$\frac{12+11+11+13+255+10+14+15+10}{9} = 39$$

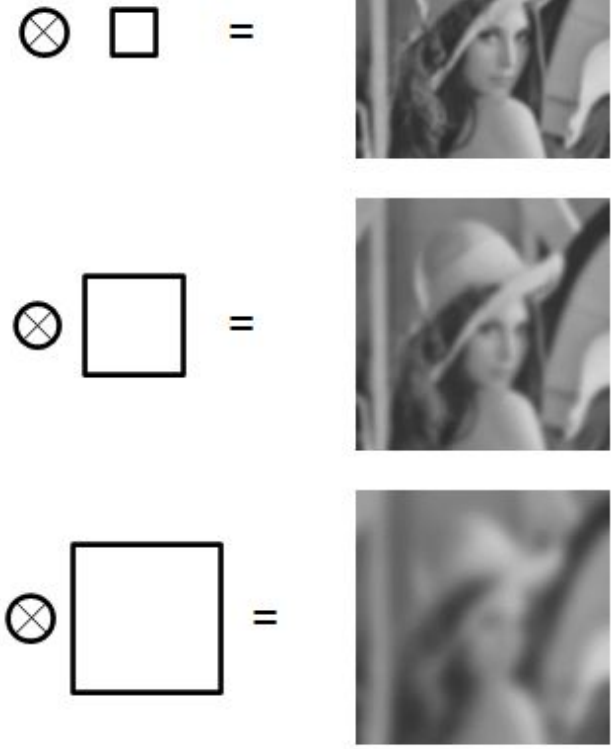
12	11	11
13	39	10
14	15	10

$$I'(u,v) = \frac{\sum_{i=-1}^{1}\sum_{j=-1}^{1} I(u+i,v+j)}{9}$$

1	1	1
1	1	1
1	1	1

Applying a linear filter across a whole image

Kernel size effect



Edge detection linear filters

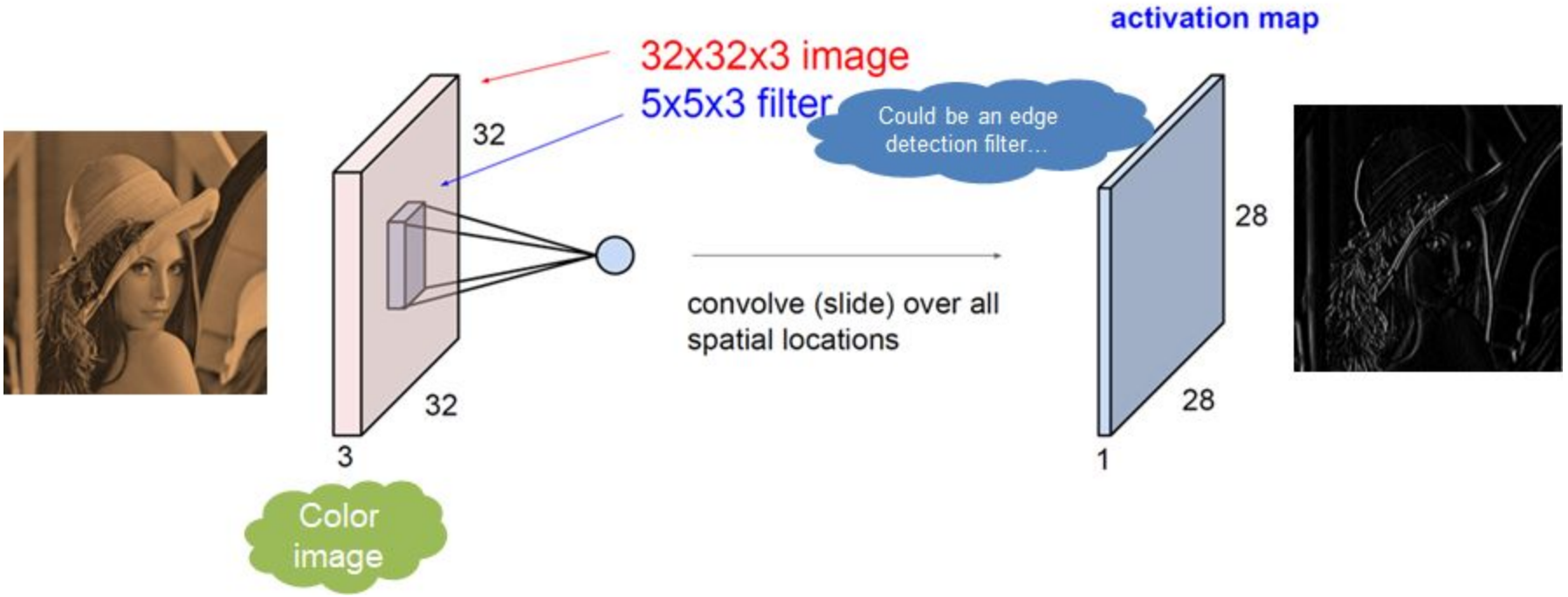
Prewitt-Filter

$$H_x^P = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad and \quad H_y^P = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

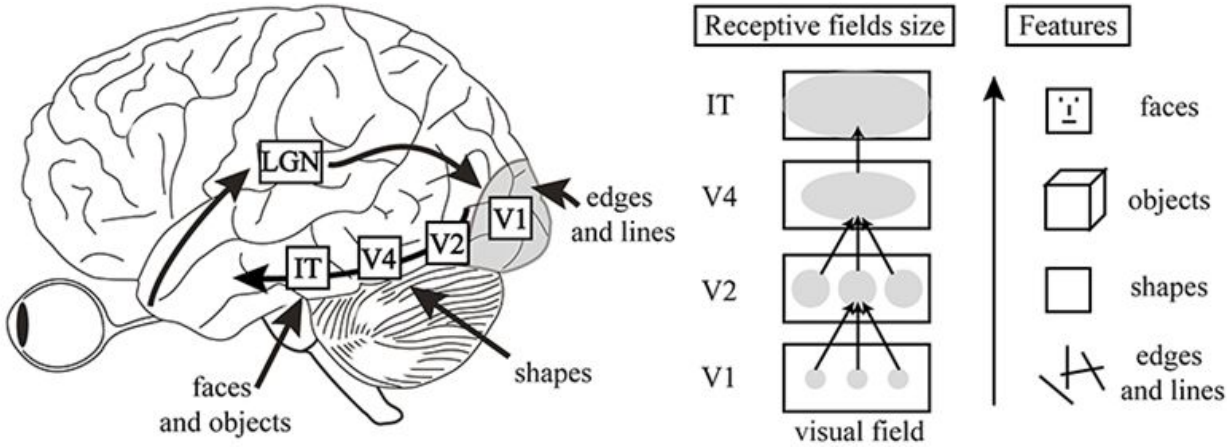
Sobel-Filter

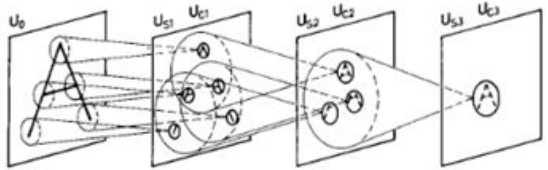
$$H_x^S = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad and \quad H_y^S = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Basis of convolutional neural networks



How does our visual cortex work?





https://neurdiness.wordpress.com/2018/05/17/deep-convolutional-neural-networks-as-models-of-the-visual-system-qa/

Convolutional NNs (1998)

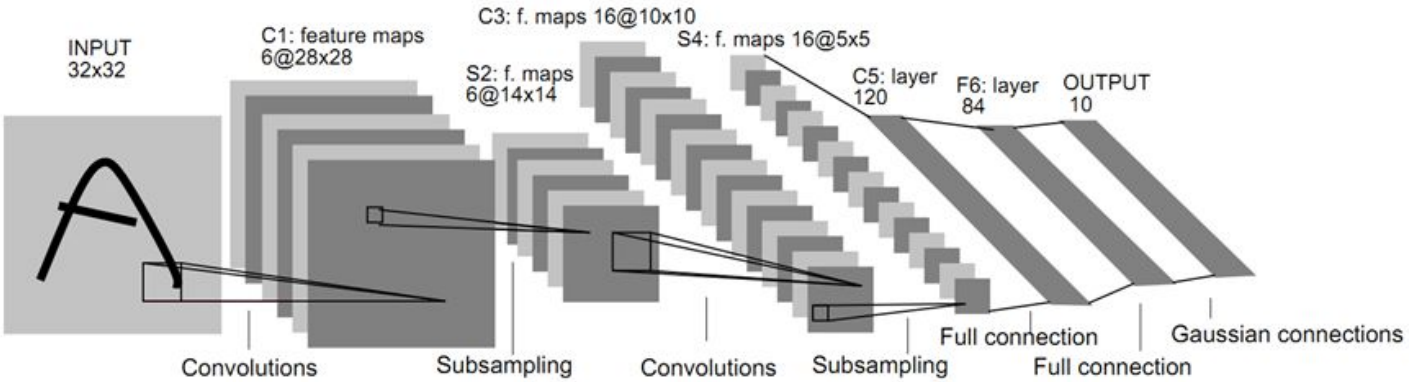


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

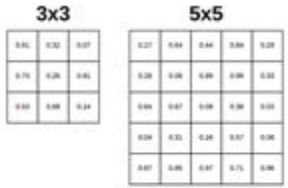
3 elements:

- Convolutions
- Subsamplings
- Fully Connected/Dense/Perceptron layers

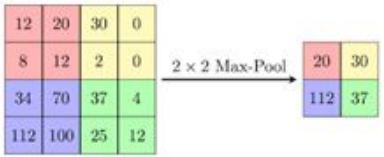
Missing the last bits....

Filters/
Activation maps

$$H_x^p = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \text{ and } H_y^p = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

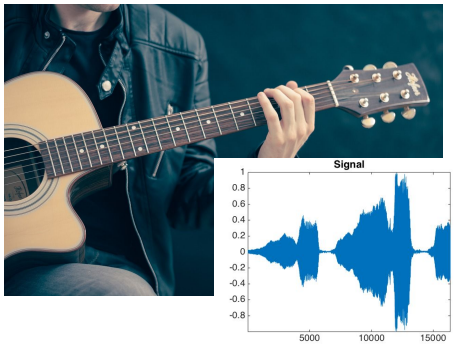


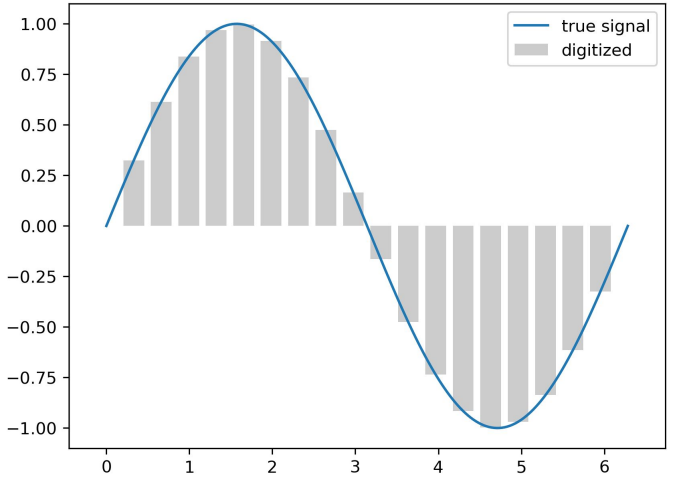

```
In [46]:
1  cnn = tf.keras.models.Sequential([
2      tf.keras.layers.Conv2D(8, (3,3), input_shape=(28,28,1), padding='same', activation='relu'),
3      tf.keras.layers.MaxPool2D(),
4      tf.keras.layers.Conv2D(16, (3,3)
5      tf.keras.layers.MaxPool2D(),
6
7      tf.keras.layers.Flatten(),
8      tf.keras.layers.Dense(128, activ
9      tf.keras.layers.Dropout(0.2),
10     tf.keras.layers.Dense(10, activation='softmax')
11 ])
```



Same as in MLPs...

Audio data



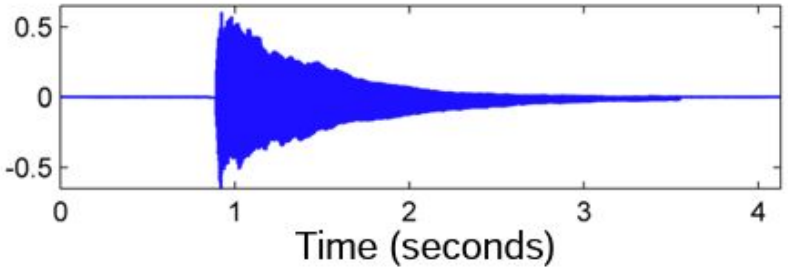


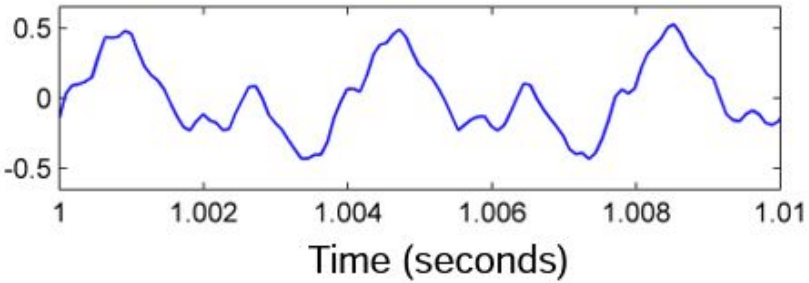
Audio signals are **raw 1D signal** with one (mono) or two channels (stereo), typically.

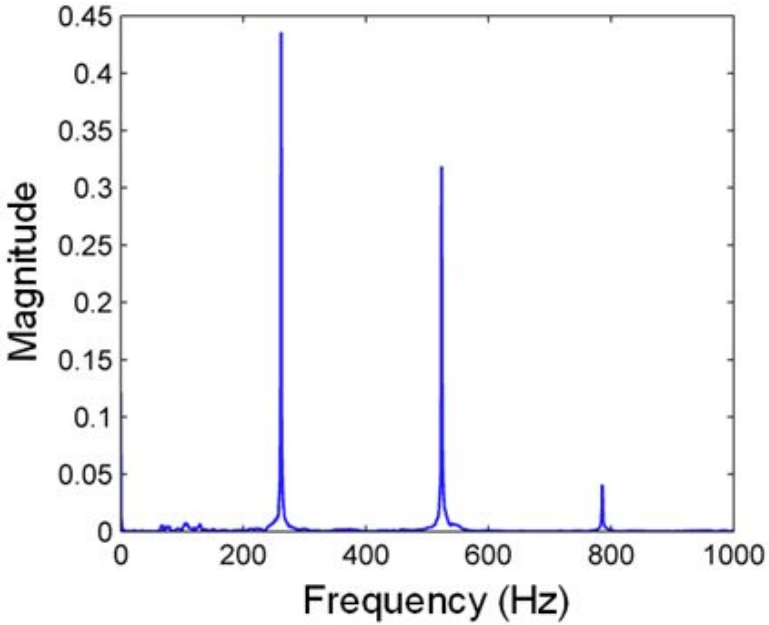
1D Convolutions for raw audio processing!

How do we get more information across time?

FOURIER SPACE!

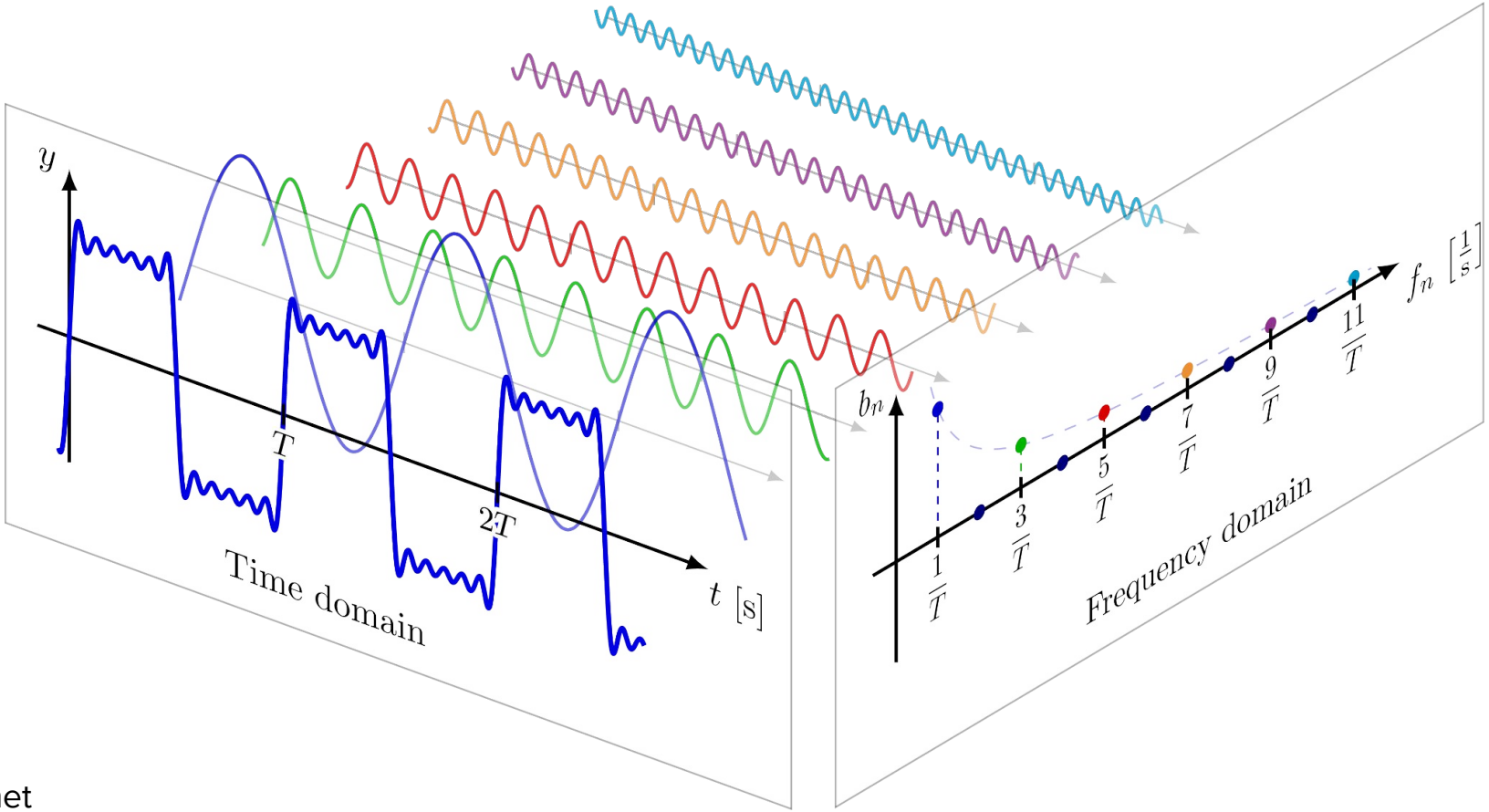






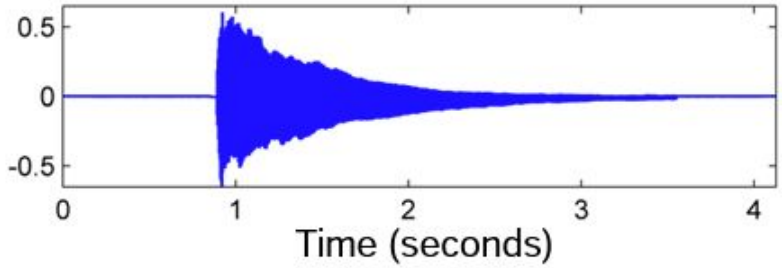
© Meinard Müller, Audio labs

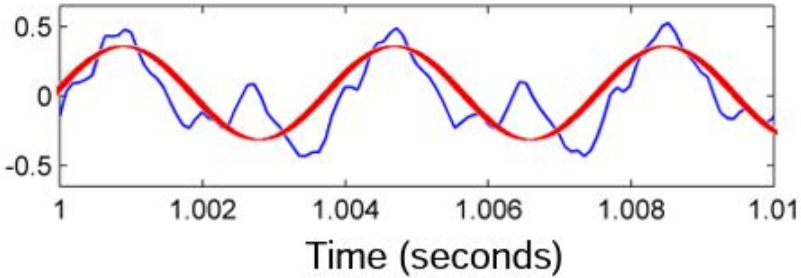
Fourier transformation ctn.

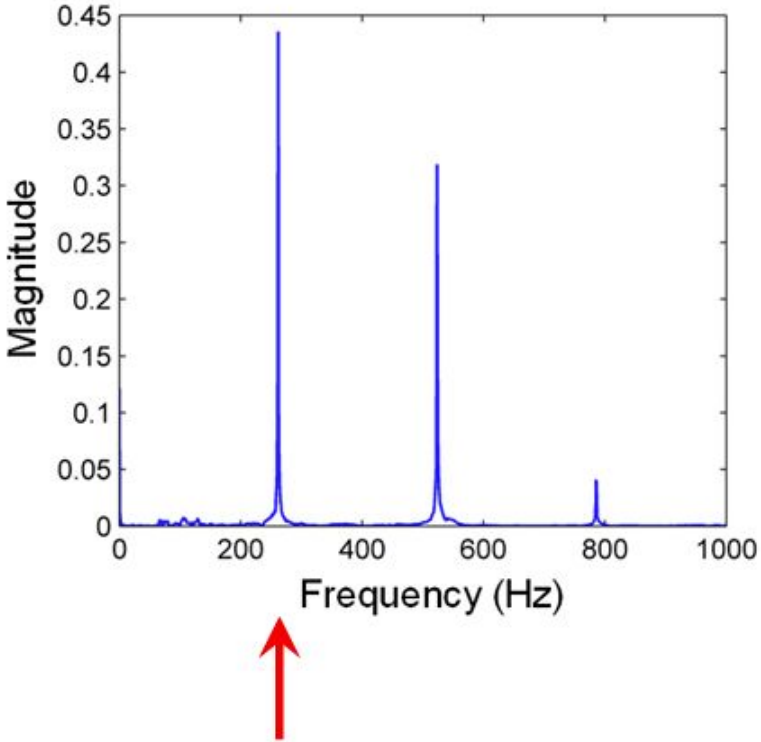


tikz.net

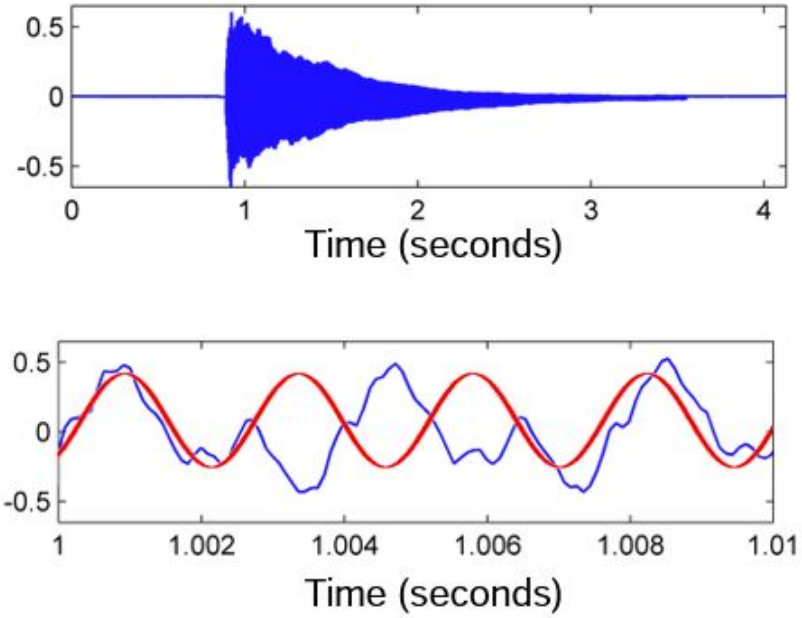
Fourier ctn.

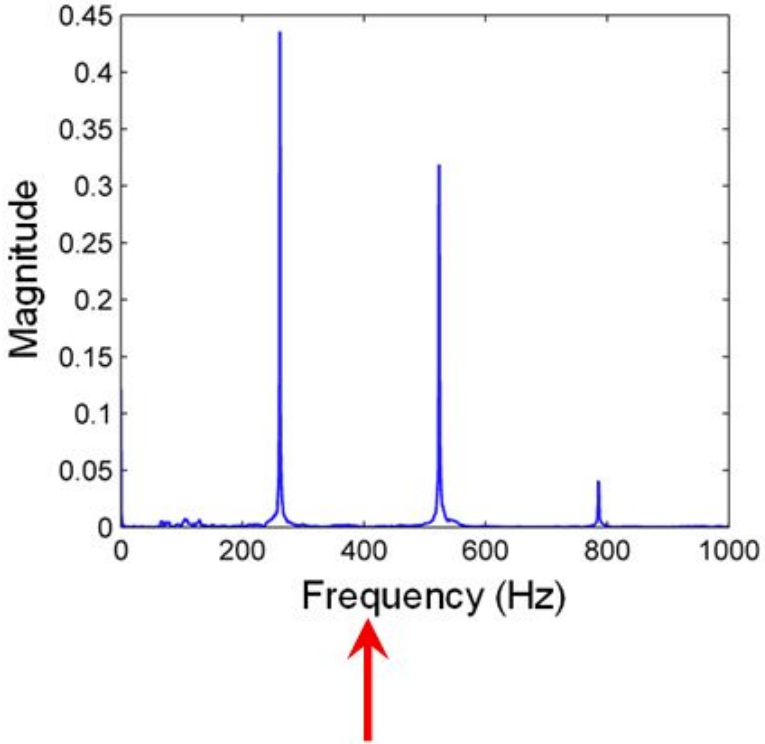

0.5
0
-0.5
0
1
2
3
4
Time (seconds)


0.5
0
-0.5
1
1.002
1.004
1.006
1.008
1.01
Time (seconds)

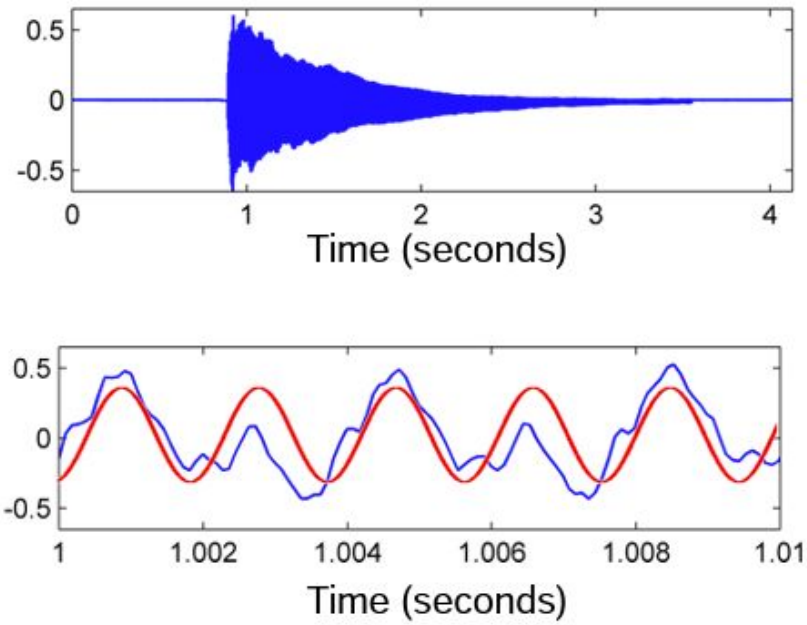

0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
Magnitude
0
200
400
600
800
1000
Frequency (Hz)

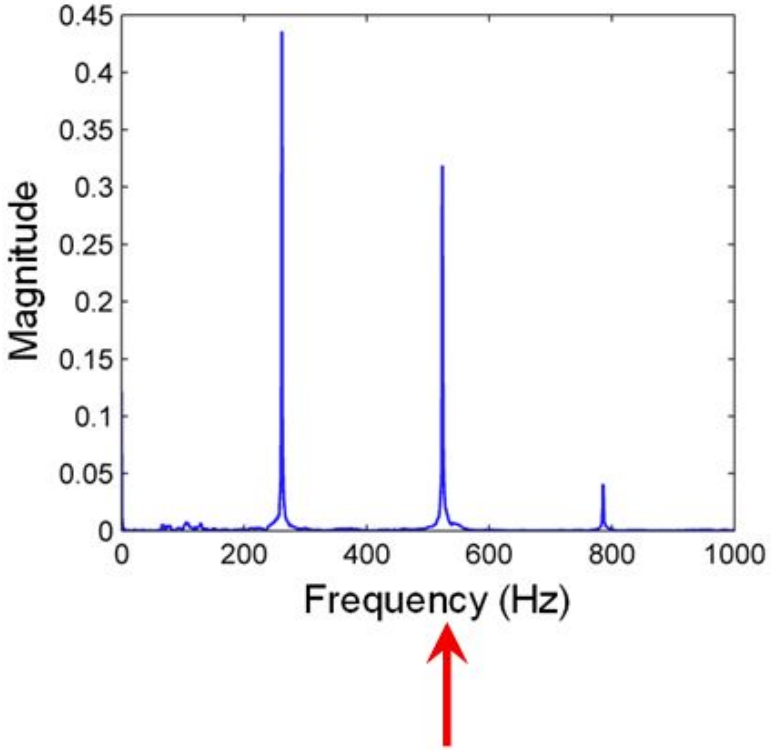
Fourier ctn.


0.5
0
-0.5
0
1
2
3
4
Time (seconds)
0.5
0
-0.5
1
1.002
1.004
1.006
1.008
1.01
Time (seconds)

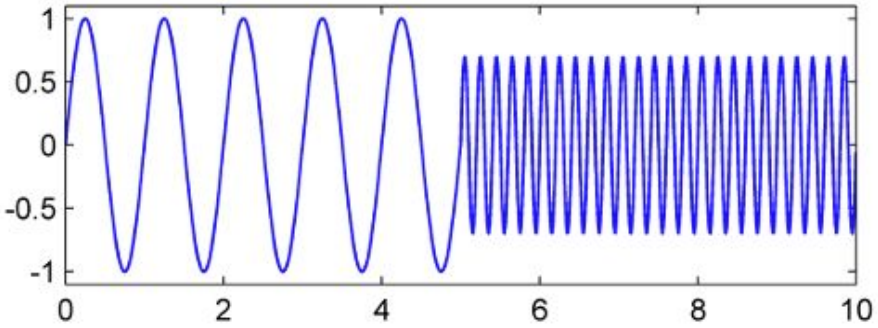

0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
Magnitude
0
200
400
600
800
1000
Frequency (Hz)

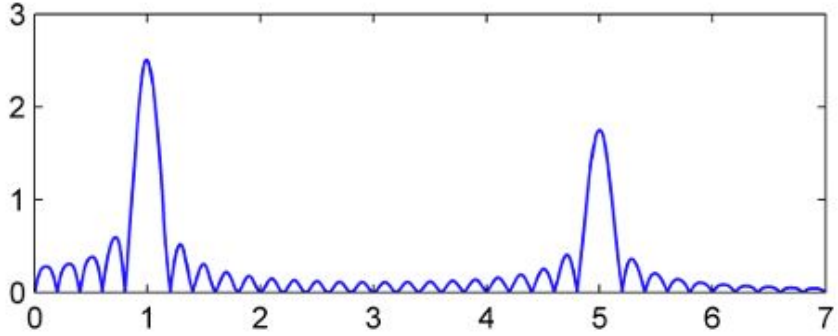
Fourier ctn.


0.5
0
-0.5
0
1
2
3
4
Time (seconds)
0.5
0
-0.5
1
1.002
1.004
1.006
1.008
1.01
Time (seconds)

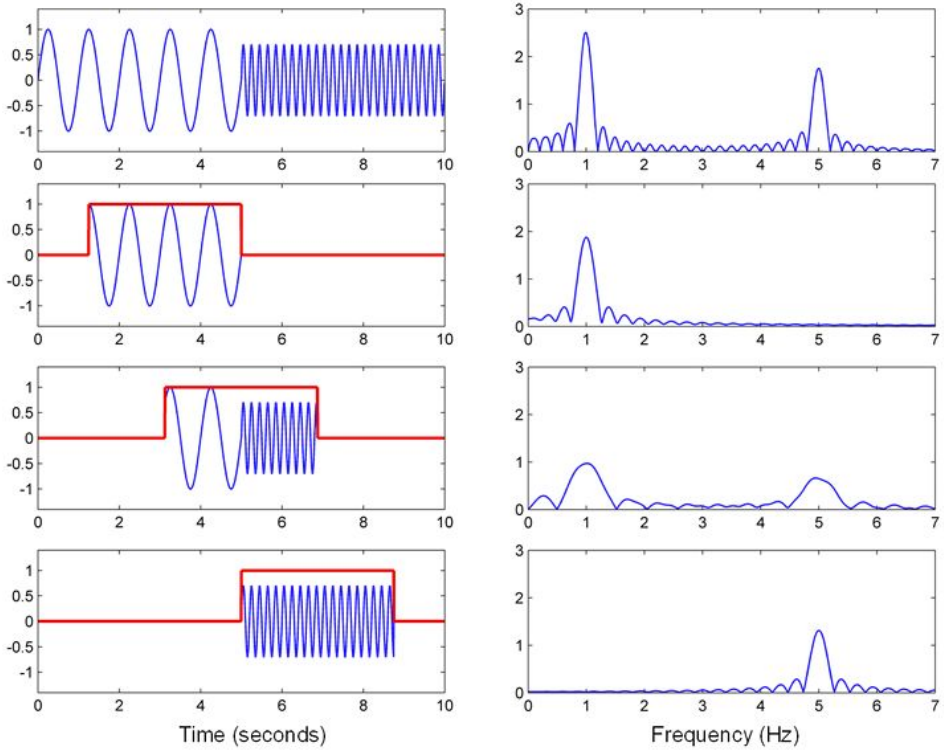

0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
Magnitude
0
200
400
600
800
1000
Frequency (Hz)

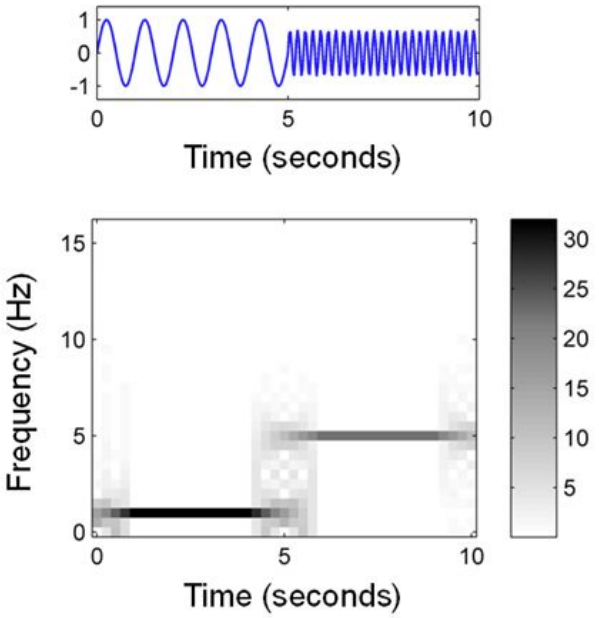
Pitch change...



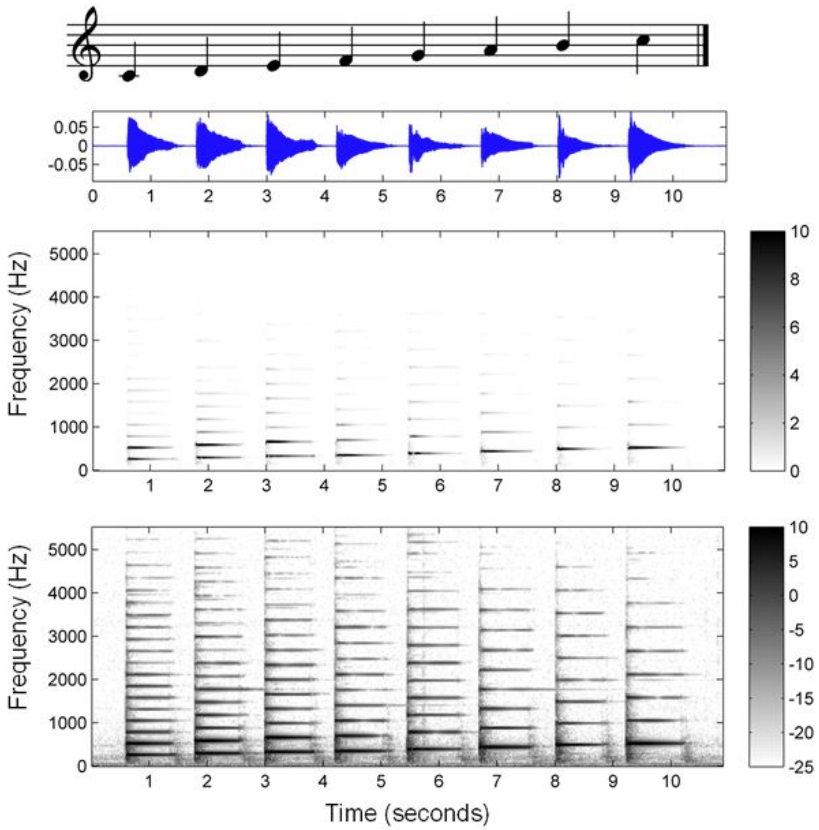


FFT across time: short-term fourier transformation (STFT)

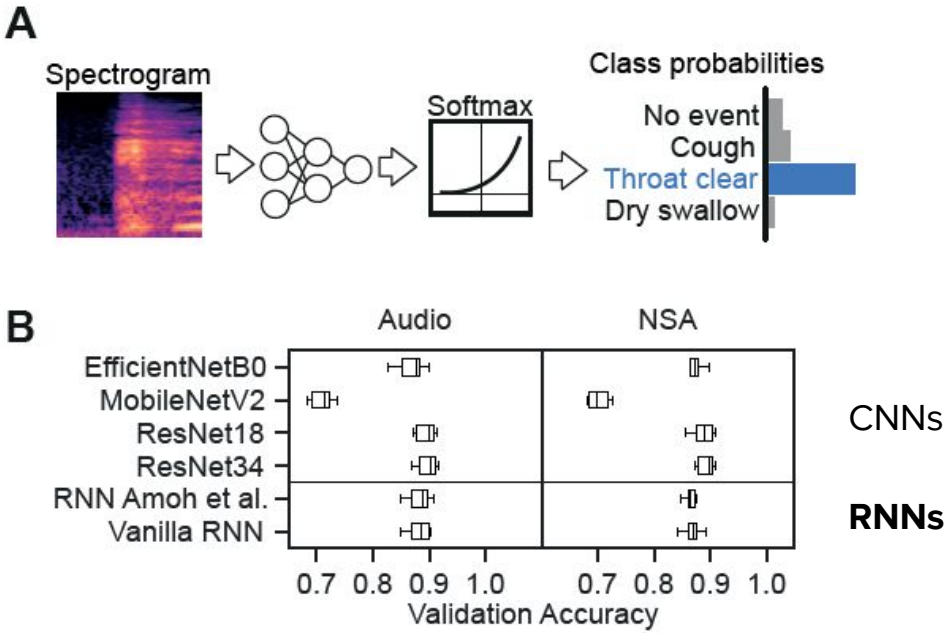




More complex input

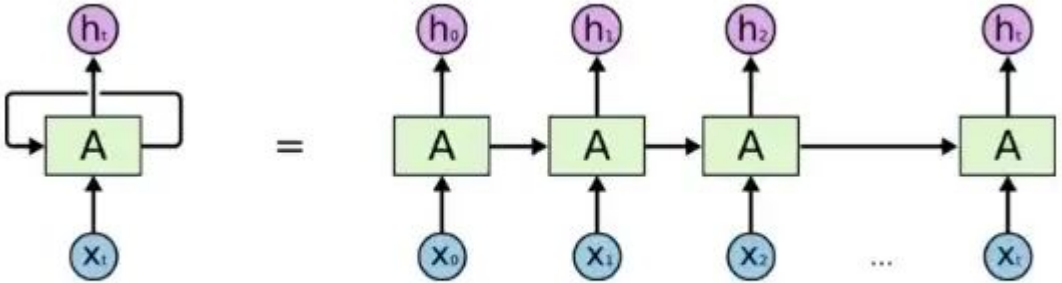


Audio input as spectogram to DNNs



Groh et al., Advanced Intelligent Systems 2022

RNNs and LSTMs



An unrolled recurrent neural network.

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t) \Rightarrow y_t = W_{hy}h_t$$

https://aditi-mittal.medium.com/understanding-rnn-and-lstm-f7cdf6dfc14e

RNNs pros and cons

Advantages of Recurrent Neural Network

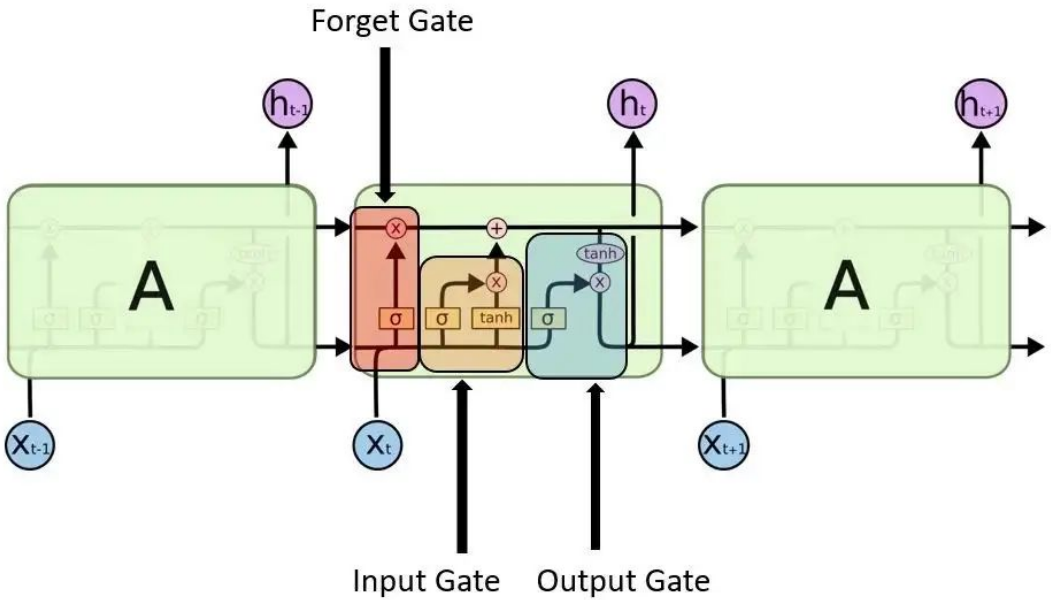
1. **RNN** can model sequence of data so that each sample can be assumed to be **dependent on previous ones**
2. Recurrent neural network are even used with convolutional layers to extend the effective pixel neighbourhood.

Disadvantages of Recurrent Neural Network

1. Gradient vanishing and exploding problems.
2. Training an RNN is a very difficult task.
3. It cannot process very long sequences if using *tanh* or *relu* as an activation function.

https://aditi-mittal.medium.com/understanding-rnn-and-lstm-f7cdf6dfc14e

Long-short term memory cells



Forget gate: decides what to keep and what to forget

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Input gate: how to shape new memory

Let through stage

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Importance stage

Output gate: How to combine input and memory to shape output

$$o_t = \sigma\left(W_o \; [\, h_{t-1}, x_t] \; + \; b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

https://aditi-mittal.medium.com/understanding-rnn-and-lstm-f7cdf6dfc14e

More time variant, and time dependent data

Fold napkins. *Polish silverware*. Wash dishes.

French napkins. *Polish silverware*. German dishes.

⇒ Context is important!

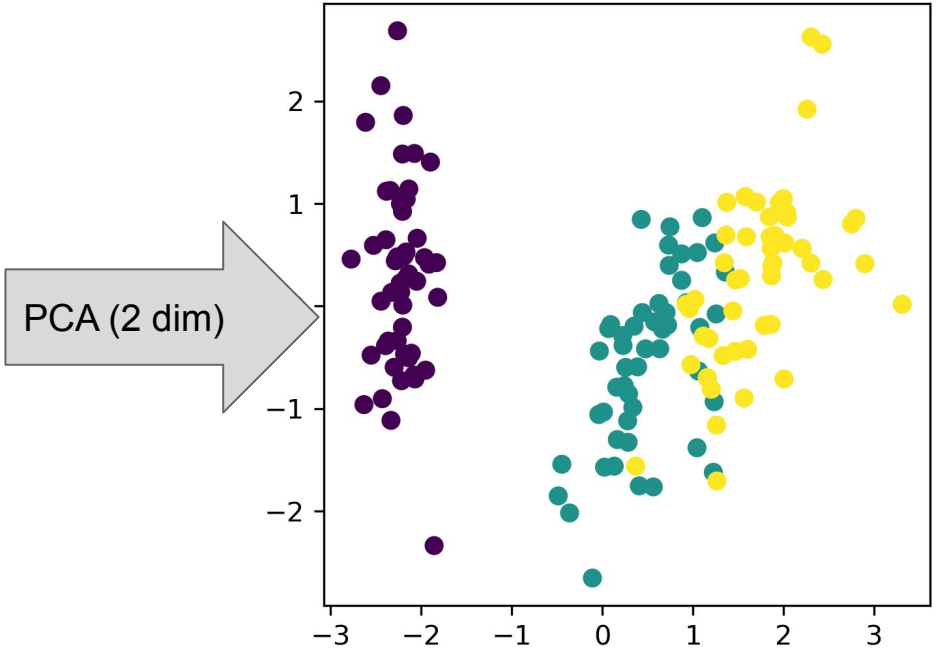
However, how to feed text towards DNNs?

Embeddings

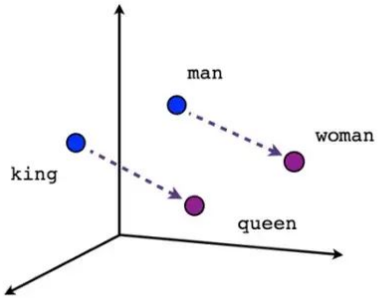
An embedding can always be created if you can compute a distance between values.

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

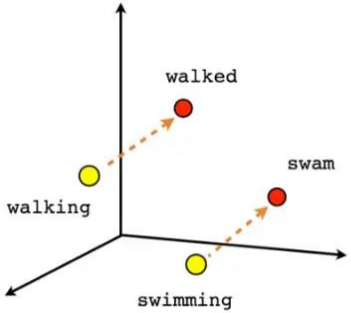
150 rows × 4 columns



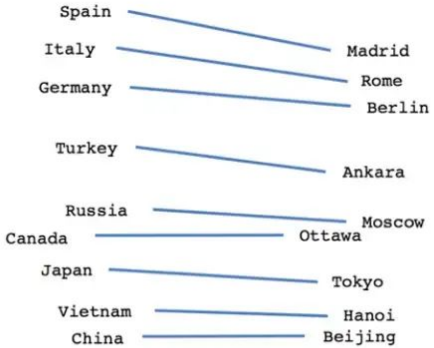
Word embeddings



Male-Female



Verb tense



Country-Capital

"**Word embedding** is the collective name for a set of language modeling and feature learning techniques in natural language processing (NLP) where **words or phrases from the vocabulary are mapped to vectors of real numbers.** The term word2vec literally translates to **word to vector**." (Wikipedia, good enough)

"dad" = [0.1548, 0.4848, ..., 1.864]

"mom" = [0.8785, 0.8974, ..., 2.794]

https://towardsdatascience.com/creating-word-embeddings-coding-the-word2vec-algorithm-in-python-using-deep-learning-b337d0ba17a8

Context!

```
The future king is the prince
Daughter is the princess
Son is the prince
Only a man can be a king
Only a woman can be a queen
The princess will be a queen
Queen and king rule the realm
The prince is a strong man
The princess is a beautiful woman
The royal family is the king and queen and their children
Prince is only a boy now
A boy will be a man
```

Window context (w=2)

The *future king* is the prince

All pairs

```
(The, future), (The, king),
(future, the), (future, king), (future, is)
(king, the), (king, future), (king, is), (king, the)
(is, future), (is, king), (is, the), (is, prince),
(the, king), (the, is), (the, prince)
(prince, is), (prince, the)
```

All pairs, cleaned

```
['future', 'king'],
['future', 'prince'],
['king', 'prince'],
['king', 'future'],
['prince', 'king'],
['prince', 'future'],
['daughter', 'princess'],
['princess', 'daughter'],
['son', 'prince']
...
```

https://towardsdatascience.com/creating-word-embeddings-coding-the-word2vec-algorithm-in-python-using-deep-learning-b337d0ba17a8

Encoding and training

```
unique_word_dict = {
 'beautiful': 0,
 'boy': 1,
 'can': 2,
 'children': 3,
 'daughter': 4,
 'family': 5,
 'future': 6,
 'king': 7,
 'man': 8,
 'now': 9,
 'only': 10,
 'prince': 11,
 'princess': 12,
 'queen': 13,
 'realm': 14,
 'royal': 15,
 'rule': 16,
 'son': 17,
 'strong': 18,
 'their': 19,
 'woman': 20
}
```

One-hot encoding:

```
'blue' = [1, 0, 0]
'car' = [0, 1, 0]
'sky' = [0, 0, 1]
```

```
a = ['blue', 'sky', 'blue', 'car']
```

```
A =
[
1, 0, 0
0, 0, 1
1, 0, 0
0, 1, 0
]
```

X= focus words

Y= context words

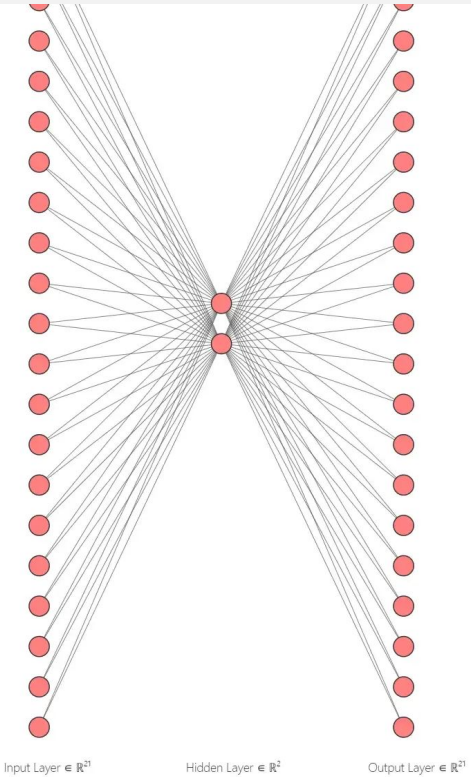
```
['future', 'king'],
['future', 'prince'],
['king', 'prince']
```

```
[array([0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.]),
 array([0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.]),
 array([0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.])]
```

```
[array([0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.]),
 array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.,
        0., 0., 0., 0.]),
 array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.,
        0., 0., 0., 0.])]
```

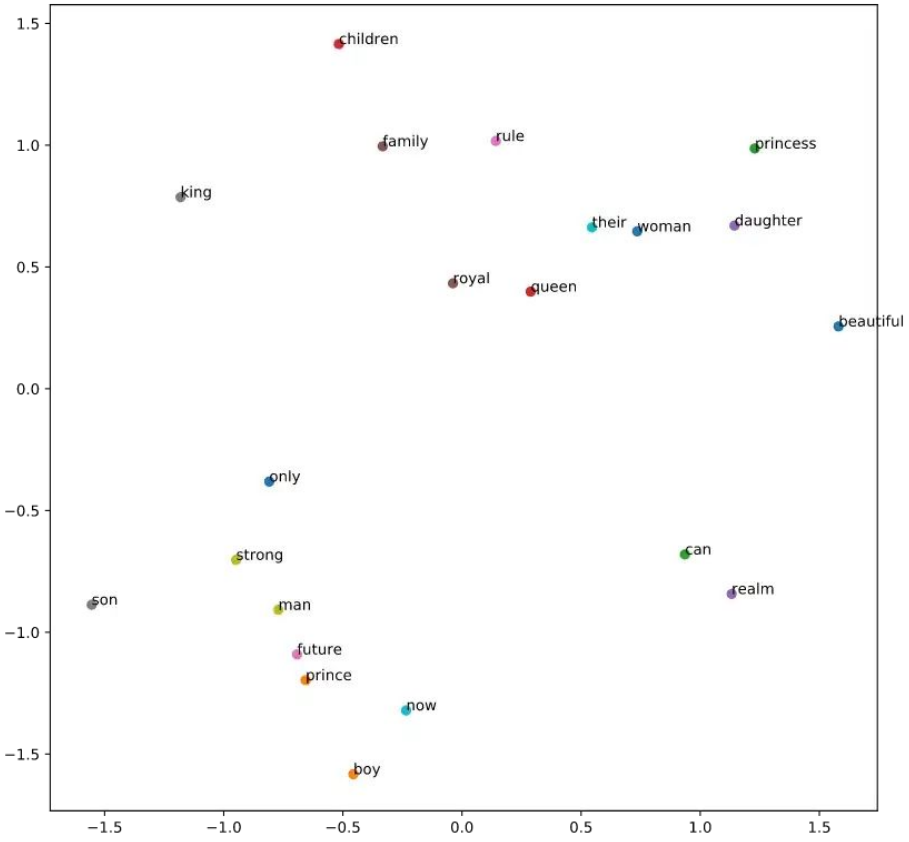
Training embedding

```
['future', 'king'],
['future', 'prince'],
['king', 'prince']
```

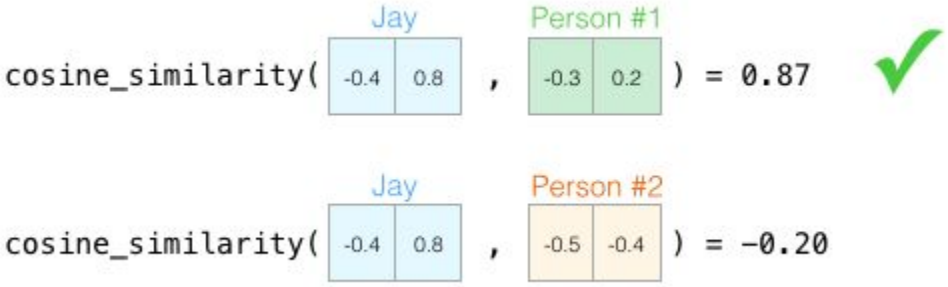


X= focus words

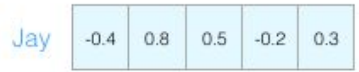
Y= context words



Similarity using distances



1- We can represent things (and people) as vectors of numbers
(Which is great for machines!)



2- We can easily calculate how similar vectors are to each other

The people most similar to Jay are:

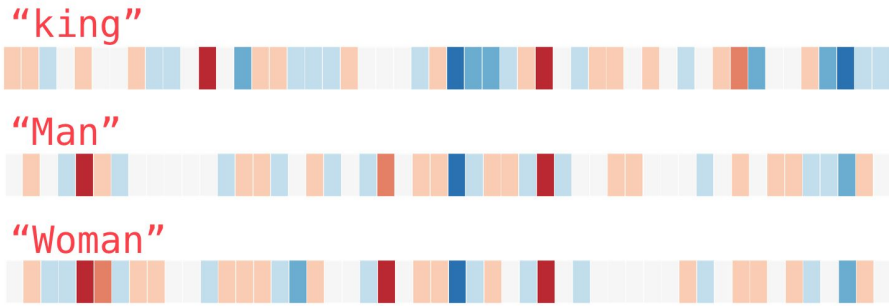
	cosine_similarity ▼
Person #1	0.86
Person #2	0.5
Person #3	-0.20

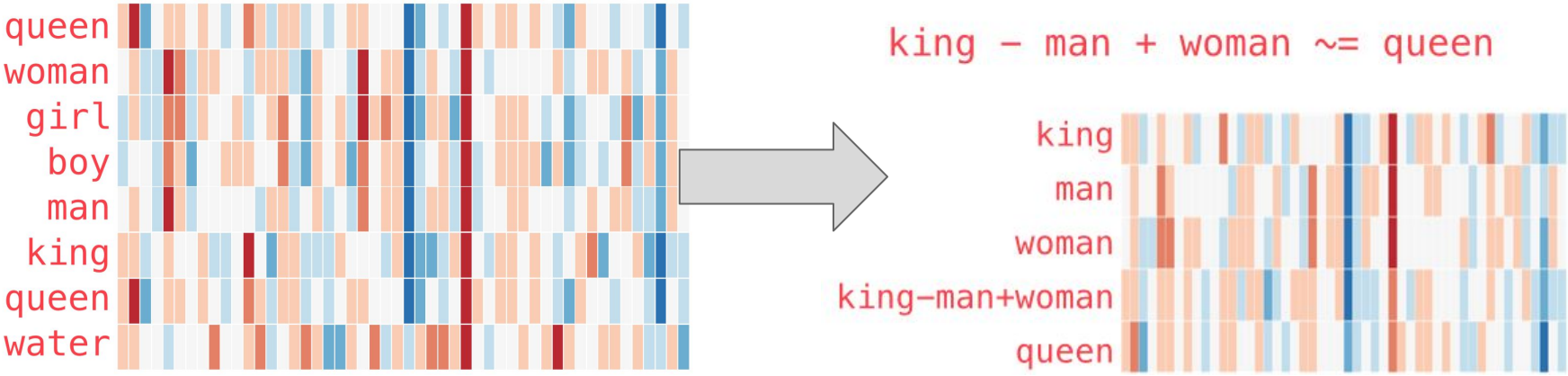
Embedding size:

2-15 rather interchangeable (also consider antonyms - similar context such as good and bad)

15-50 similarity rather defines relatedness

Example for logic





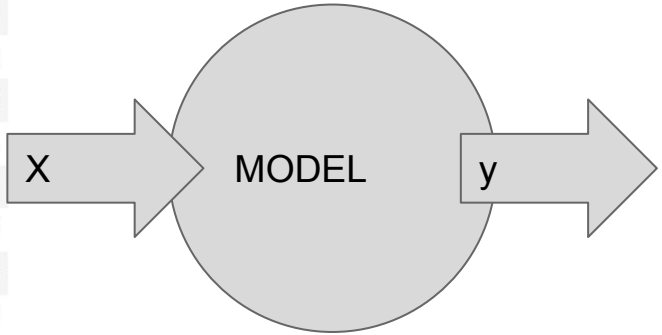
Part III: Model evaluation

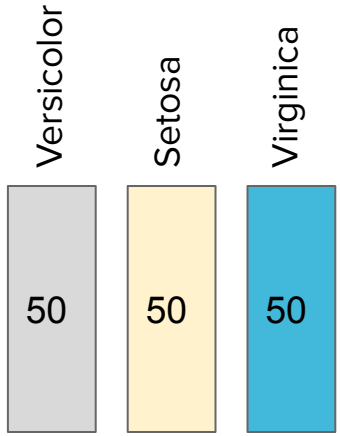
Iris dataset

Baseline accuracy: 33%!

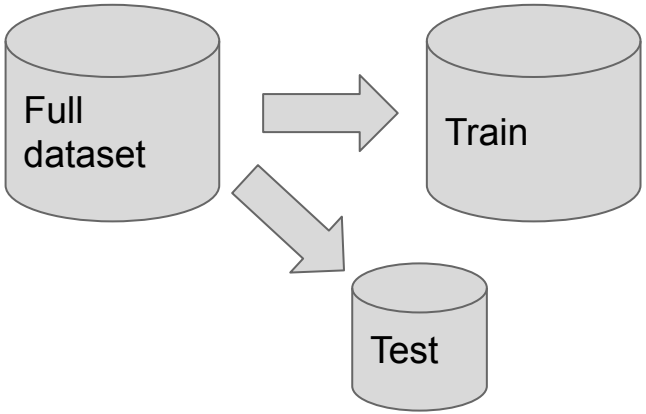
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns





Okay, can we determine if the model learnt?



- training set → 38 × Setosa, 28 × Versicolor, 34 × Virginica
- test set → 12 × Setosa, 22 × Versicolor, 16 × Virginica

i.i.d. We assume that the training examples are i.i.d (independent and identically distributed), which means that all examples have been drawn from the same probability distribution and are statistically independent from each other. A scenario where training examples are not independent would be working with temporal data or time-series data.

⇒ STRATIFICATION of your data
Allowing equal sampling of classes

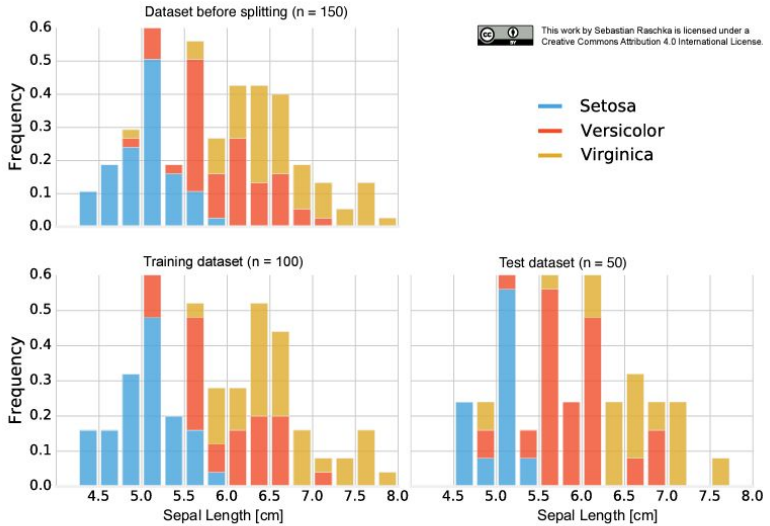


Figure 1: Distribution of *Iris* flower classes upon random subsampling into training and test sets.

Raschka 2020, arXiv:1811.12808v3

Holdout validation

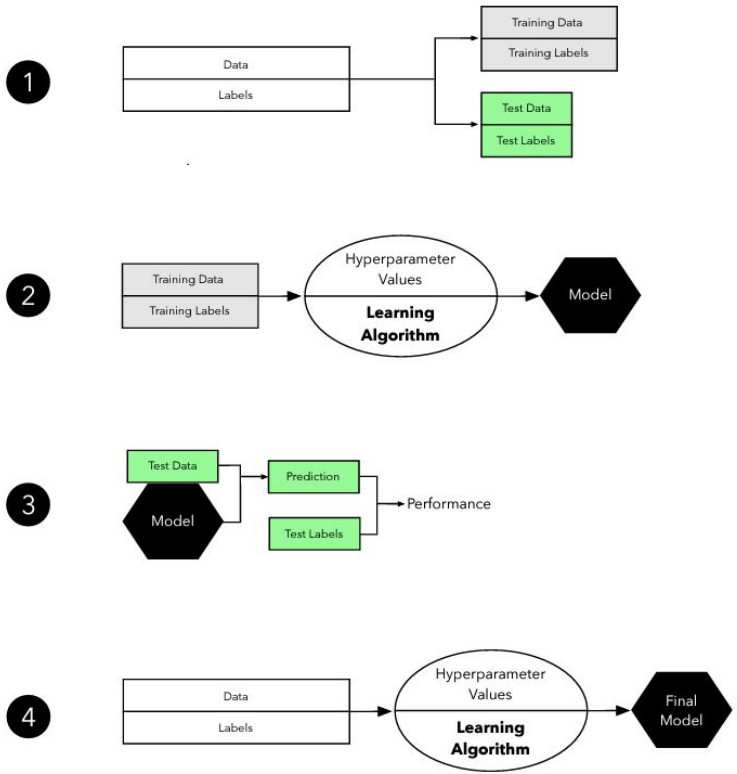
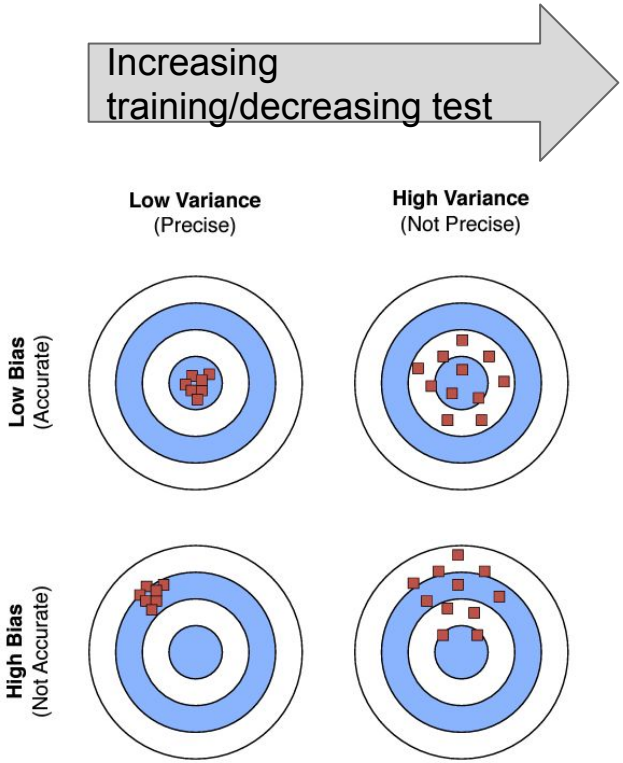


Figure 2: Visual summary of the holdout validation method.

Raschka 2020, arXiv:1811.12808v3



Training = test set ⇒ optimistically biased, test set too large in respect to training set ⇒ pessimistically biased

Bootstrapping

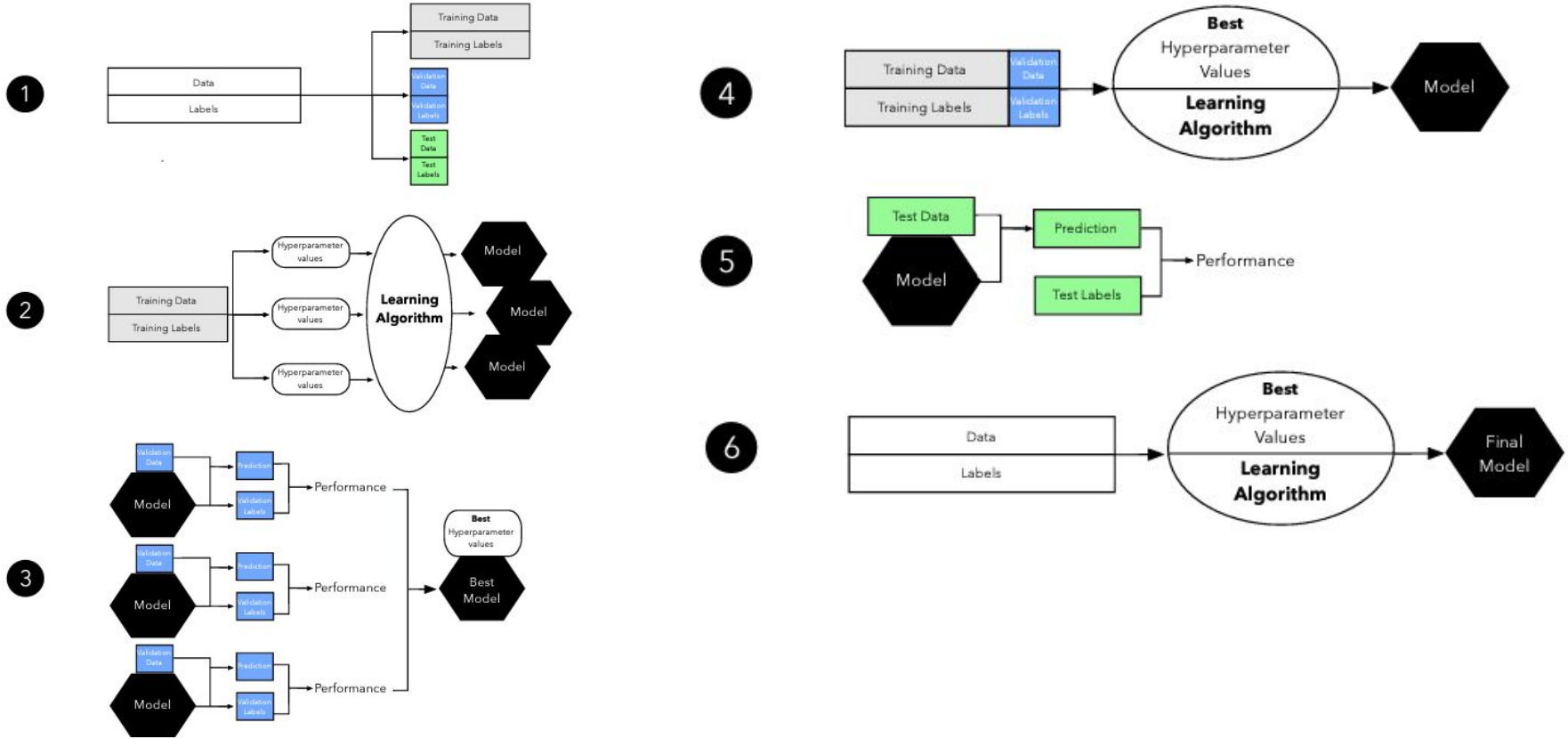
Sampling/drawing always **from the original dataset!**

Figure 7: Illustration of training and test data splits in the Leave-One-Out Bootstrap (LOOB).

50-200 bootstrap samples should be good
[Efron and Tibshirani, 1994]

Raschka 2020, arXiv:1811.12808v3

Hyperparameter tuning/model selection



Raschka 2020, arXiv:1811.12808v3

K-fold crossvalidation

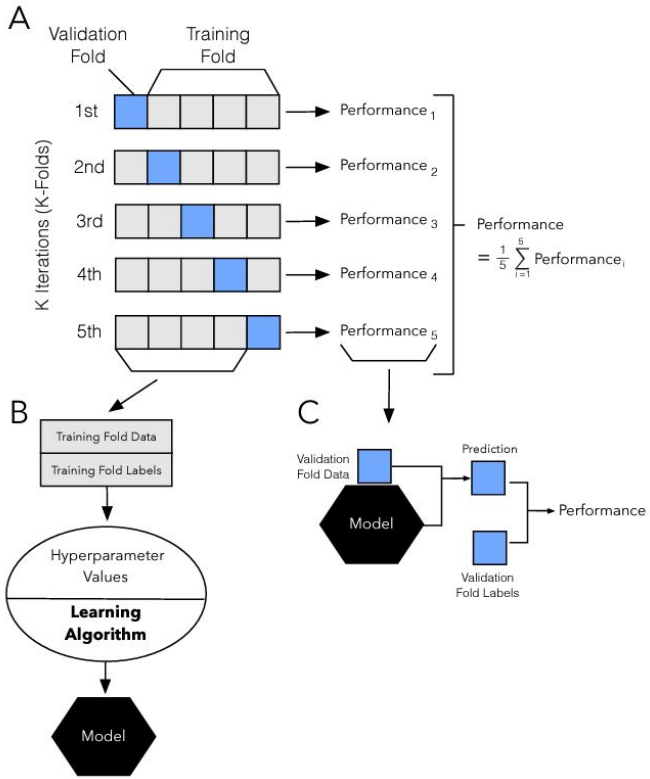


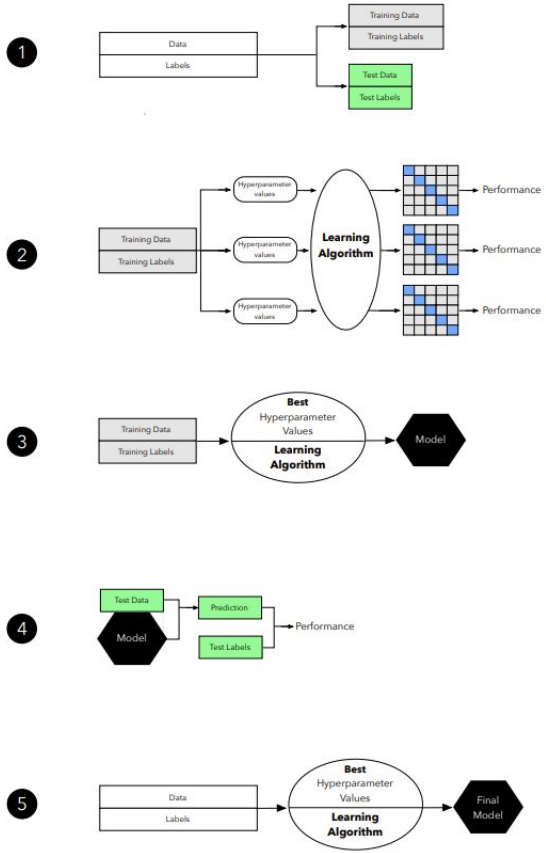
Figure 13: Illustration of the k-fold cross-validation procedure.

Leave-one-out-Val.

Figure 15: Illustration of leave-one-out cross-validation.

Raschka 2020, arXiv:1811.12808v3

Model selection w/ k-fold crossval

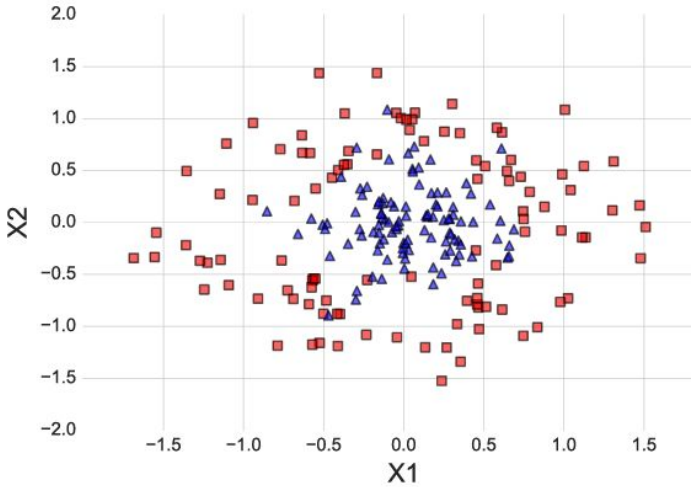


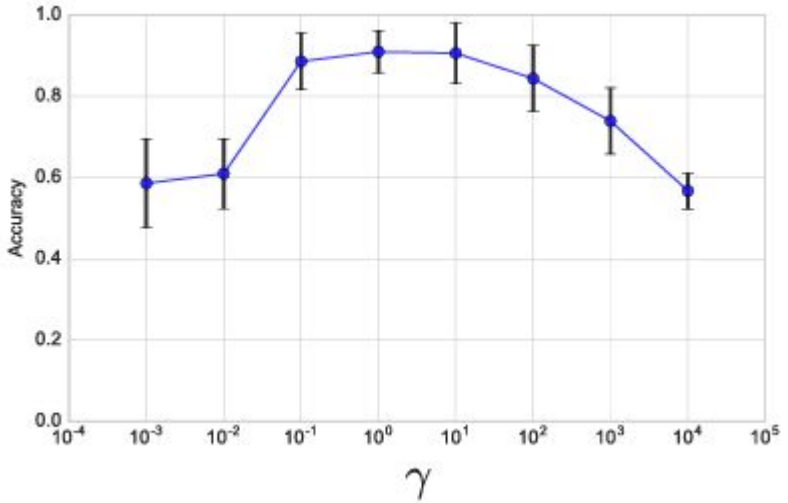
Raschka 2020, arXiv:1811.12808v3

Example: tuning hyperparameter Gamma in SVMs

Now, let us assume that our goal is to optimize the γ (gamma) hyperparameter of a Support Vector Machine (SVM) with a non-linear Radial Basis Function-kernel (RBF-kernel), where γ is the free parameter of the Gaussian RBF:

$$K(x_i, x_j) = exp(-\gamma||x_i - x_j||^2), \gamma > 0. \quad (46)$$

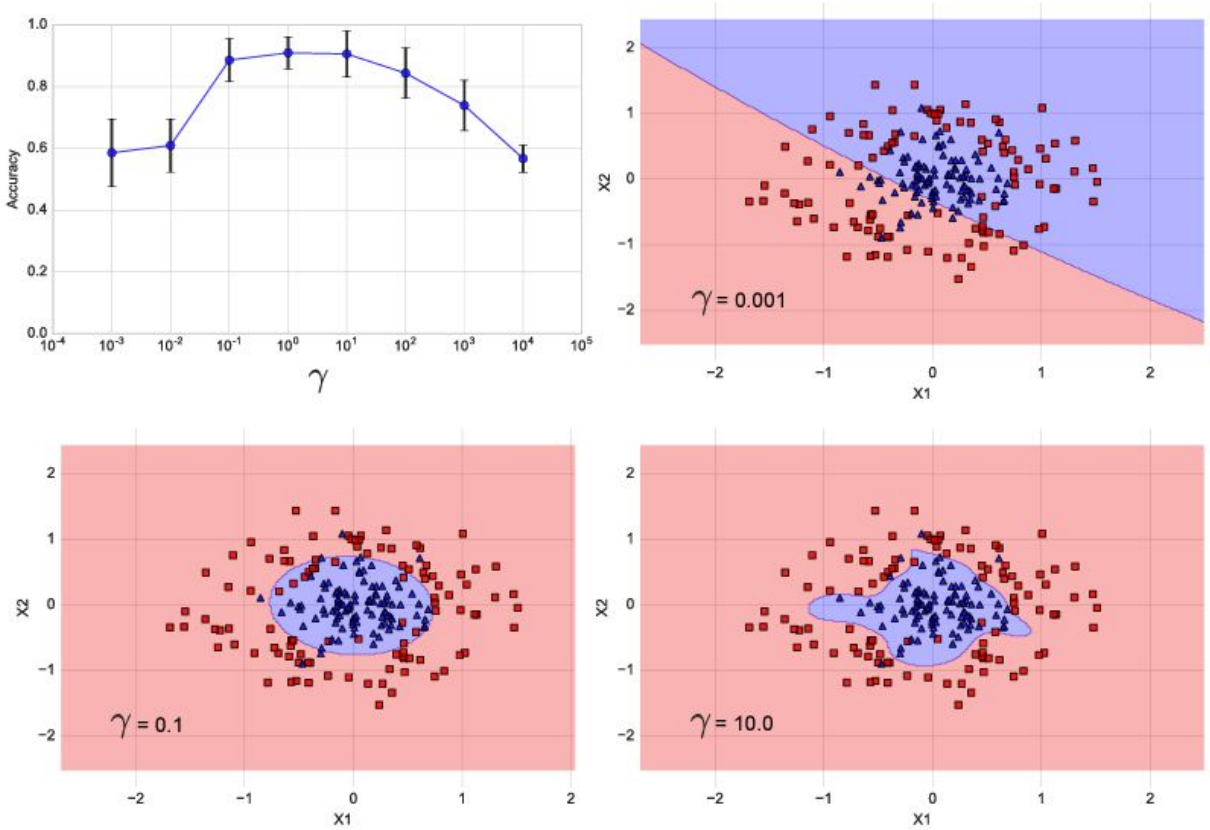




10-fold crossvalidation, mean performance, error bars indicate std.

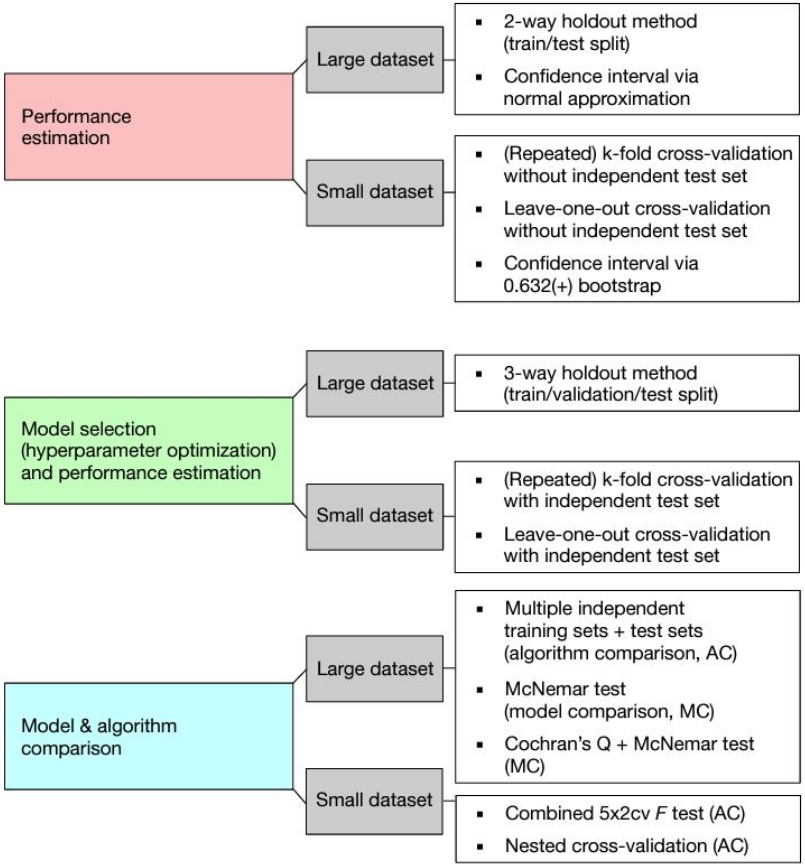
Raschka 2020, arXiv:1811.12808v3

Looking at Gamma in detail



Raschka 2020, arXiv:1811.12808v3

Summary



Raschka 2020, arXiv:1811.12808v3

Homework

In this homework, you will build, train and evaluate a Convolutional Neural Network using TensorFlow.

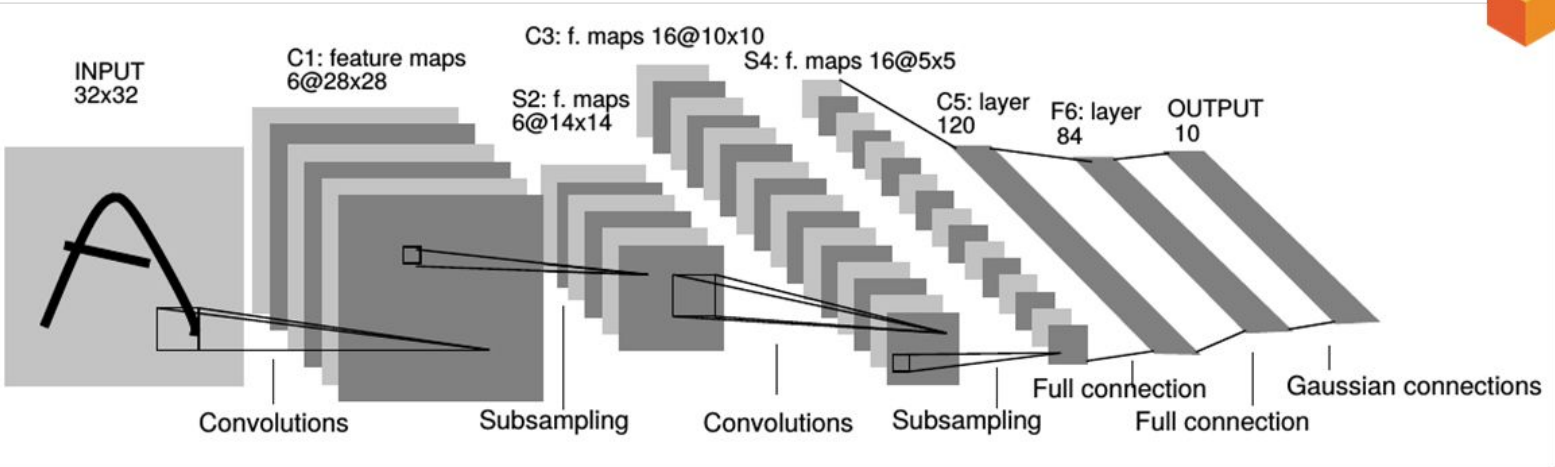


Fig. 2. Architecture of LeNet-5, a convolutional NN, here used for digits recognition. Each plane is a feature map, i.e., a set of units whose weights are constrained to be identical.