# Deep Learning – Notas de Aula

Sumário

[Deep Learning – Notas de Aula 1](#_Toc58163720)

[Unidade 1 – As Origens do Deep Learning 3](#_Toc58163721)

[Inteligência Artificial 3](#_Toc58163722)

[IA – Principais Aplicações 3](#_Toc58163723)

[IA – Áreas de Atuação 3](#_Toc58163724)

[Aprendizado de Máquina (ML) 3](#_Toc58163725)

[Cenários de Aprendizagem 3](#_Toc58163726)

[Python e Machine Learning 7](#_Toc58163727)

[Redes Neurais 7](#_Toc58163728)

[Unidade 2 – Fundamentos de Deep Learning 11](#_Toc58163729)

[Deep Learning 11](#_Toc58163730)

[Áreas de Pesquisa do Deep Learning 11](#_Toc58163731)

[Origem do Deep Learning 12](#_Toc58163732)

[Principais Componentes do Deep Learning 13](#_Toc58163733)

[Playground 16](#_Toc58163734)

[Deep Learning - Como Funciona? 16](#_Toc58163735)

[Principais Frameworks 16](#_Toc58163736)

[Algoritimo - Redes Neurais 17](#_Toc58163737)

[Algoritimo de Processamento de Texto 17](#_Toc58163738)

[Algoritimo de Processamento de Imagem 17](#_Toc58163739)

[Unidade 3 – Redes Convolucionais, Recorrentes e Adversárias 18](#_Toc58163740)

[Redes Neurais Convolucionais 18](#_Toc58163741)

[Algoritmo – Operação Convolucional 19](#_Toc58163742)

[Arquitetura CNN 19](#_Toc58163743)

[Algoritmo – Classificação de Dígitos 21](#_Toc58163744)

[Regularização 21](#_Toc58163745)

[Algoritmo – Classificação de Imagens 21](#_Toc58163746)

[Redes Neurais Recorrentes 22](#_Toc58163747)

[Algoritmo – RNN 22](#_Toc58163748)

[Processamento de Texto 23](#_Toc58163749)

[Algoritmo – Geração de Texto 24](#_Toc58163750)

[Redes Generativas Adversárias (GAN) 24](#_Toc58163751)

[Unidade 4 - Considerações Finais 27](#_Toc58163752)

[Transferência de Aprendizado 27](#_Toc58163753)

[Algoritimo - Transferência de Aprendizado 27](#_Toc58163754)

[Revisão - Deep Learning 27](#_Toc58163755)

[Revisão - Deep Learning - Arquiteturas e Aplicações 28](#_Toc58163756)

[Algoritimo de Classificação de Cães e Gatos 30](#_Toc58163757)

[Algoritimo de Classificação de Cães e Gatos – Resposta 30](#_Toc58163758)

[IA, ML, DL e Ciência de Dados 30](#_Toc58163759)

[Aplicando o Conhecimento – Exemplos de Aplicações 30](#_Toc58163760)

# Unidade 1 – As Origens do Deep Learning

## Inteligência Artificial

* Inteligência Artifical > Aprendizado de Máquina > Deep Learning
* IA simbólica: IA programada com lógica condicional. É uma inteligência estática, que não evolui.
* Surgimento de IA ocorreu na segunda guerra mundial, para auxiliar na análise balística e quebra de criptografias do inimigo.
* Teste de Turing: Um computador pode pensar?

## IA – Principais Aplicações

* Aplicações: Jogos, reconhecimento de fala, robótica, processamento de linguagem natural, sistemas especialistas (identificação de spam, planejamento logístico), veículos autônomos, visão computacional.
* IA é uma tecnologia da moda. Há uma hype muito grande nas empresas por soluções de IA, pois isso dá uma ideia de empresa inovadora aos clientes, além das empresas estarem sempre em busca de elementos que aumentem a sua competitividade.

## IA – Áreas de Atuação

* Aprendizado de Máquina e reconhecimento de padrões
* Motores de Busca (Google)
* Heurística: Trabalha por aproximação. Não retorna a melhor resposta, mas é uma solução mais rápida (menos custosa).
* IA simbólica, baseada em lógica
* Planjemaneto: Uso em linha de produção, foco em maior retorno e menor custo.
* Programação genética: Seleção dos mais aptos.

## Aprendizado de Máquina (ML)

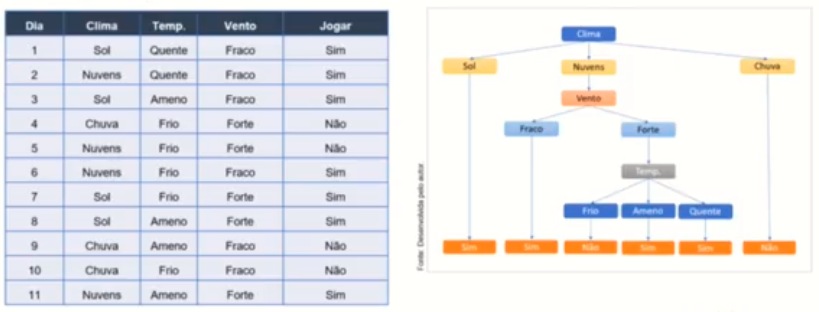
* Utiliza experiência adiquirida para melhorar seu conhecimento
* Tensor: matriz multidimensional, com linhas, colunas e outras dimensões.
* Técnicas de Machine Learning
  + Classificação: Atribuição de rótulos aos dados
  + Regressão: Previsão de valores
  + Ranqueamento: Utilizado nos sistemas de busca. Ranqueamento realizado com base em critérios pré-estabelecidos.
  + Agrupamento: Separação dos dados em grupos homogêneos.
  + Redução de Dimensionalidade / Aprendizado Múltiplo
* Função de Perda: Mede a diferença, ou perda, entre o rótulo previsto e o rítulo real no conjunto de testes.

## Cenários de Aprendizagem

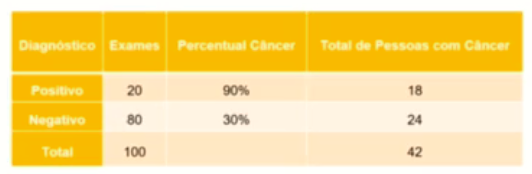
* Aprendizado supervisionado: o sistema recebe um conjunto de dados já rotulados para a fase de treinamento e faz previsões para os demais dados não conhecidos. É o cenário mais comum ao se tratar problemas de classificação, regressão e ranqueamento. Possui dados de treinamento.
  + Exemplo: Identificando emails de SPAM. Nós informamos ao algoritmo um conjunto de emails que são SPAM e outro que não é SPAM. Esses são os dados de treinamento. O algoritmo aprende e é validado, e pode classificar emails futuros como SPAM ou não SPAM.
  + Exemplo: Classificação de emails em categorias no GMail. O algoritmo classifica os emails nas categorias “Principal” (emails pessoais), “Social” (emails de redes sociais), “Promoções” (emails de marketing, descontos e ofertas), “Atualizações” (atualizações pessoais, contas, extratos bancários, recibos) e “Fóruns” (mensagens de grupos online e listas de distribuição). As categorias foram previamente treinadas e testadas. Depois que conseguiram classificar corretamente as mensagens, foram colocadas em produção aos usuários do Gmail.
  + Exemplo: Uso de regressão linear para prever o valor de um imóvel. Treinar o algoritmo com diferentes imóveis com suas caraterísticas e o preço deles. Depois novos imóveis terão seu valor previsto.

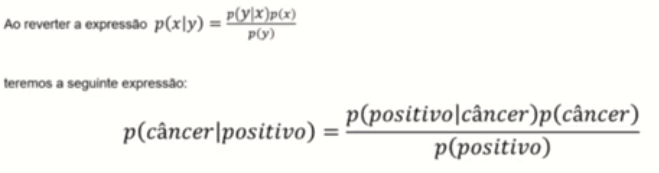


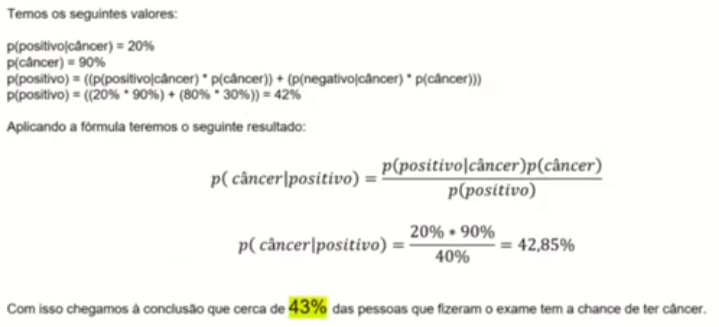
* + Exemplo: Árvore de decisão. Algoritmo é treinado com um conjunto de características e decisões e depois gera uma árvore para tomar uma decisão com base em novos dados de entrada.



* + Exemplo: Algoritmo de Naive Bayes. Trabalha com probabilidade, é simples, fácil de implementar, é capaz de processar grandes volumes de dados e tem um tem um bom tempo de resposta.

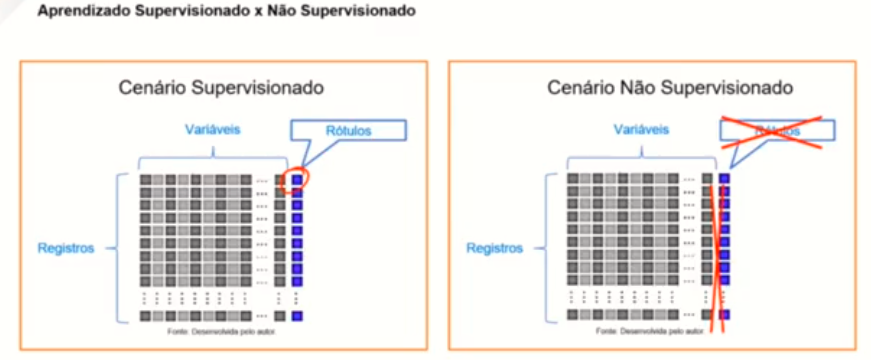




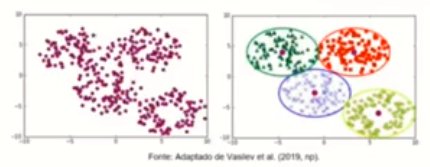


* Aprendizado não supervisionado: Recebe um conjunto de dados não rotulados, e faz previsão para todos eles. Visto que não existem dados rotulados, é difícil inferir a performance do aprendizado do algoritmo.
  + Problemas como agrupamento e redução de dimensionalidade são comumente resolvidos utilizando aprendizado não supervisionado.



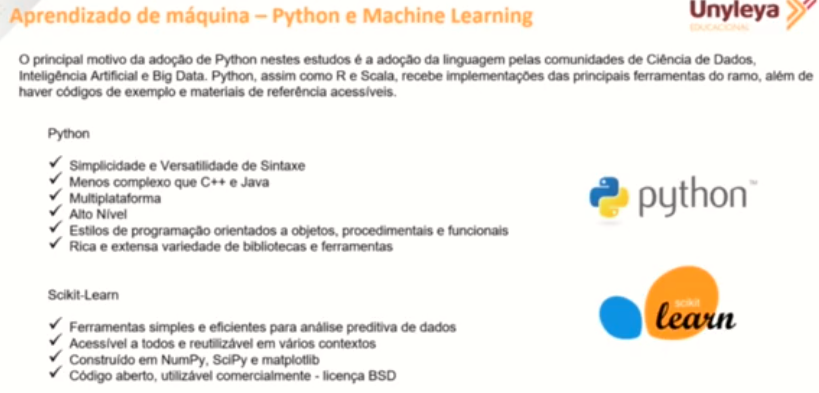


* + Exemplo de uso: Classificação de emails entre SPAM e não-SPAM. Na abordagem não supervisionada o algoritmo não sabe o que é um SPAM e um email normal, pois não tem dados de treinamento.
    - (1) Separação de emails entre SPAM e não-SPAM;
    - (2) Combinação e similaridade das amostras;
    - (3) Métricas diferentes para clusters diferentes;
    - Nem sempre precisa especificar o número de clusters.
  + Exemplo: Utilizado em Deep Learning para processamento de linguagem natural (PLN). Como funciona:
    - Alimentação das palavras na rede neural;
    - Vetor numérico para cada palavra (word2vec);
    - Càlculos através dos valores numéricos.
  + Exemplo: Utilização do algoritmo K-Means. Ele realiza o agrupamento de elementos em K clusters, K sendo o número de clusters.



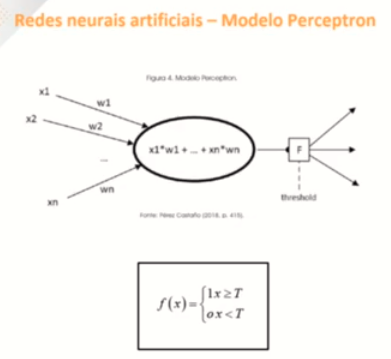
* Aprendizado semi-supervisionado: O sistema recebe um conjunto de treinamento com dados rotulados e não rotulados e faz previsões para todos os dados desconhecidos.
* Inferência Transdutiva: O sistema recebe para treinamento um conjunto de dados rotulados juntamente com o conjunto de teste sem os respectivos rótulos.
* Aprendizado online: Ao contrário dos outros cenários, este envolve múltiplas rodadas e as fases de treinamento e teste são mescladas.
* Aprendizado reforçado: As fases de treinamento e teste se misturam e, para coletar informações, o sistema interage ativamente com o ambiente e, em alguns casos, interfere no ambiente, e recebe uma recompensa por cada ação.
* Aprendizado ativo: O sistema coleta, de forma adapatativa ou interativa, exemplos de treinamento, geralmente consultando um guia para solicitar rótulos para novos dados.

## Python e Machine Learning

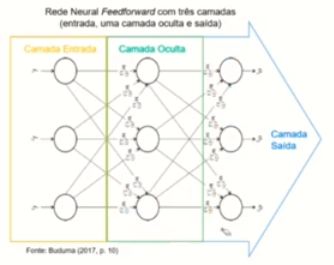


## Redes Neurais

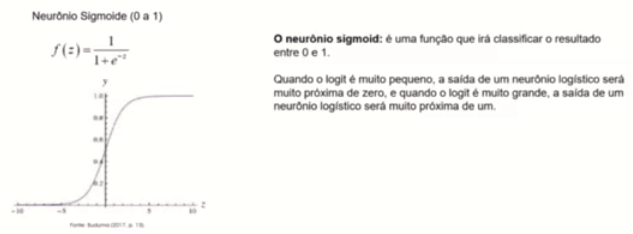
* Neurônio artifical: Recebe valores de entrada, transforma valores e transmite valores a outros neurônios.
* Modelo Perceptron: É um classificador linear, ou seja, uma rede neural que não possui camadas ocultas.
  + Recebe diversas entradas.
  + Calcula a soma ponderada das entradas e pesos.
  + Aplica limite (threshold).
  + Envia para próxima camada



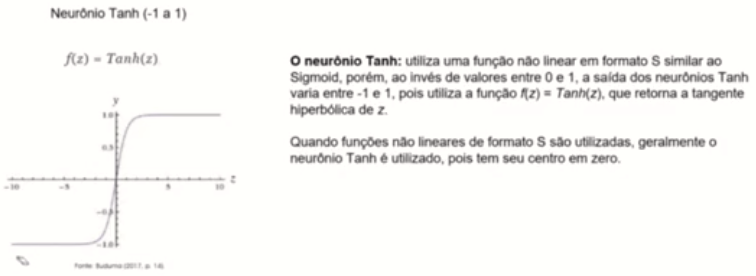
* Redes neurais artificais – feedforward
  + Organizado em camadas;
  + Neurônios na mesma camada não se comunicam;
  + Possui uma ou mais camadas.



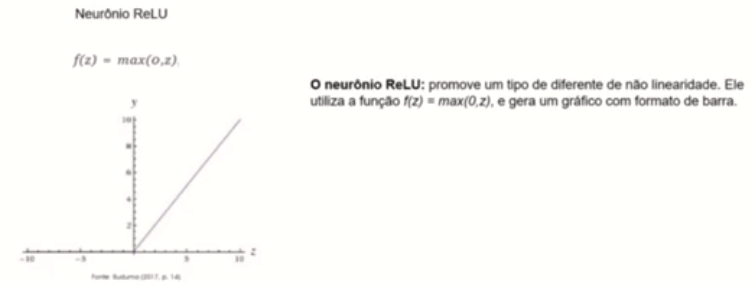
* A camada oculta possui esse nome porque não é possível prever a saída desejada nas camadas intermediárias.
* Neurônios especiais, funções de ativação: Neurônios Sigmoid, Tanh e ReLU.
  + Neurônio Sigmoid: É uma função que irá classificar o resultado entre 0 e 1.



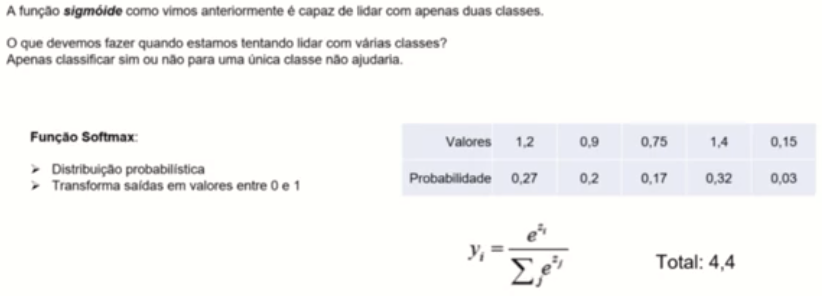
* + Neurônio Tanh (tangente): Utiliza uma função não linear em formato S. A saída é entre -1 e 1.



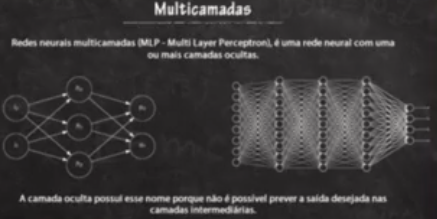
* + Neurônio ReLU: Resultado em formato de reta.



* Camada de saída softmax: A função sigmóide só é capaz de trabalhar com duas classes (0 ou 1). Já a função softmax pode trabalhar com uma série de classes. Ele calcula a probabilidade dos valores de cada classe.



* Processo de treinamento
  + Cuidados
    - Treine seu algoritmo com dados diversificados;
    - Como o ser humano, a rede neural aprende com erros e acertos;
    - Maior o conjunto de treinamento, melhor o resultado;
    - Qualidade no conjunto de treinamento auxilia no resultado final;
    - Parametrização correta no treinamento da rede neural.
  + Minimize os erros com técnicas de
    - Gradiente Descente
      * Determina melhor vetor de pesos
      * Busca o erro zero
    - Backpropagation
      * Otimização de pesos
      * Seleção de melhores entradas e saídas
* Multi-layer Perceptron (Multicamadas)
  + Quanto as redes neurais possuem mais de uma camada, elas são chamadas de Redes Neurais Profundas (**Deep Neural Network**), que são comumente utilizadas em **Deep Learning**.



# Unidade 2 – Fundamentos de Deep Learning

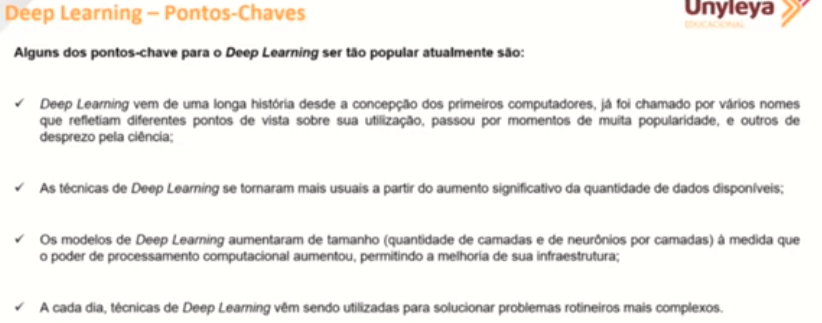
## Deep Learning

* O que define um problema de Deep Learning
  + Problemas de IA complexos, nos quais os sistemas precisam adquirir seu próprio conhecimento.
  + Problemas com muitas variações nas características dos dados, que apresentam processo de tratamento complicado.
  + Problemas que não conseguem ser solucionados com técnicas de Machine Learning.
* Como funciona uma solução de Deep Learning
  + (1) Problema é quebrado em partes menores
  + (2) Cada parte menor possui características mais simples para análise
  + (3) Transforma conceitos complexos em conceitos simples
* Tipos de modelos de Deep Learning
  + Deep Learning Feedforward / Multicamadas / Multilayer Perceptron (MLP): Função matemática que mapeia dados de entrada em saídas, e é composta por diversas funções mais simples.
* Por que utilizar uma rede neural?
  + Melhor aprendizado da representação dos dados
  + Permite o computador aprender, passo a passo, a cada camada
  + Não existe uma quantidade exata de camadas que determina se uma rede é ou não de aprendizado profundo
* Deep Learning – conceito: Pode ser considerado como o estudo de modelos que envolvem uma grande quantidade de funções de aprendizado que levam ao aprendizado de máquina.

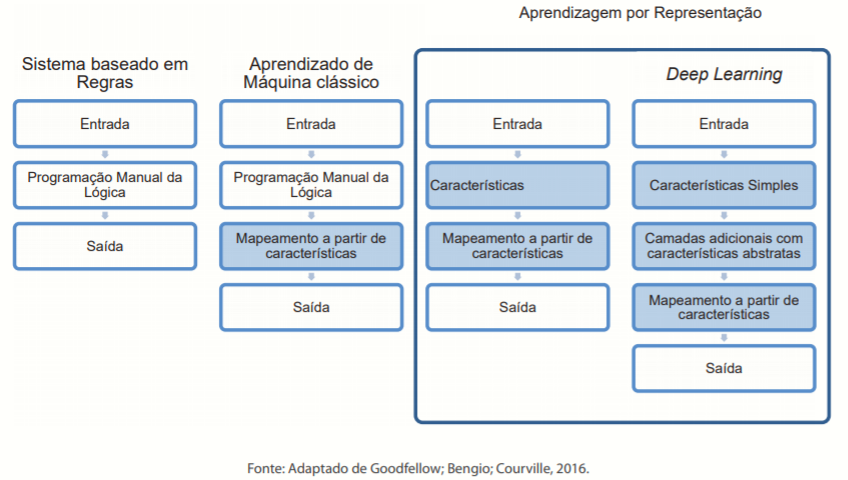
## Áreas de Pesquisa do Deep Learning

* Visão Computacional
  + Reconhecimento de objetos, pessoas, animais, gatos e cães, insetos.
* Reconhecimento e classificação de Fala (reconhecimento fonético, busca por voz, reconhecimento de conversa, classificação de semântica etc.)
  + Reconhecimento da pessoa
  + Transcrição para texto
  + Assistente pessoal
* Entendimento de Linguagem Natural
  + Classificação de texto (Livro, notícia, artigo)
  + Resumo de texto
  + Identificação do gênero do livro
* Robótica
  + Casa – Aspirador robô
  + Indústria – Automação na linha de montagem
  + Exército – Exoesqueletos
  + Polícia/Bombeiros – Desarmamento de bomba
* Análise de moléculas e descoberta de novos medicamentos
  + Identificação de novas doenças e novos medicamentos

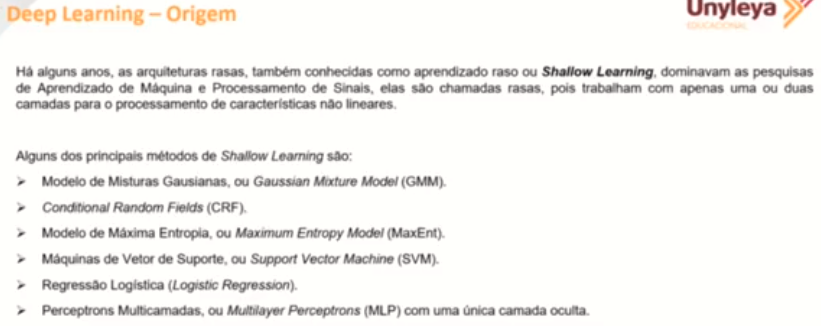
## Origem do Deep Learning



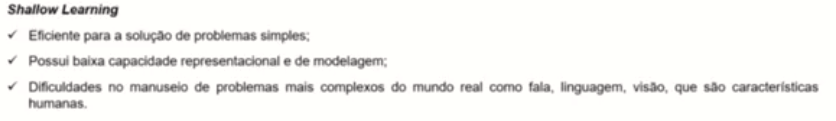
* Diferentes tipos de IA



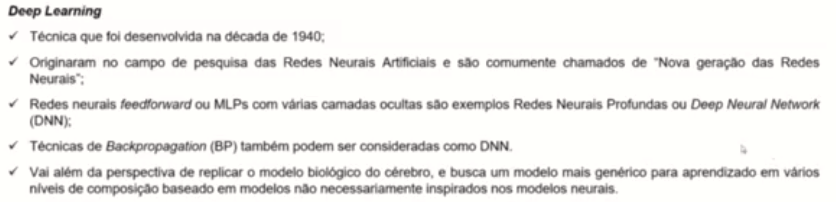
* Origem do Deep Learning



* Shallow Learning



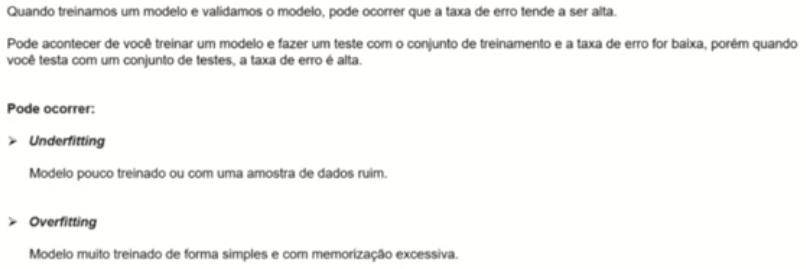
* Necessidade de novas técnicas para resolver problemas complexos: Deep Learning



* Deep Learning passou por três grandes fases, quando assumiu os seguintes nomes:
  + I) Cibernética, entre as décadas de 1940 e 1960;
  + II) Conexionismo entre as décadas de 1960 a 1990;
  + III) Deep Learning, a partir da década de 2000.
  + A mudança do nome da técnica ao longo das técnicas fez com que fosse perdida a rastreabilidade da técnica no tempo, até que recentemente foi chamada de Deep Learning e ganhou popularidade.

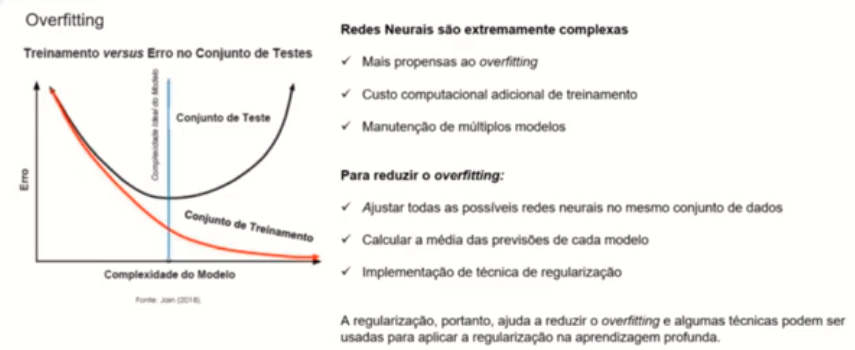
## Principais Componentes do Deep Learning

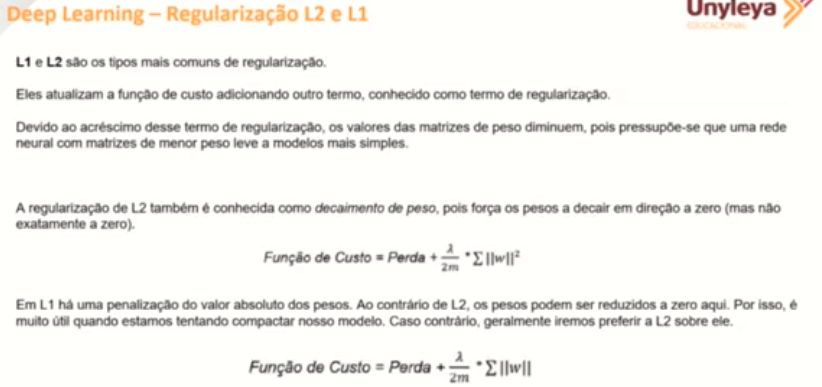
* Componentes básicos
  + Parâmetros
  + Camadas
  + Funções de ativação
  + Funções de perda
  + Métodos de otimização
  + Hiperparâmetros
* Parâmetros: Estão diretamente relacionados aos pesos nas conexões da rede
* Camadas: Temo as as camadas de entrada, camadas ocultas e camadas de saída em uma rede neural que podem ser ajustadas através de hiperparâmetros.
* Funções de ativação
  + Função de Ativação para Arquiteturas Gerais
  + Funções de Ativação para Camadas Ocultas
  + Camadas de Saída para Regressão
  + Camadas de Saída para Classificação Binária
  + Camadas de Saída para Classificação Multiclasse
  + Camada de Saída para Classificações Múltiplas
* Funções de perda / custo
  + Quantifica a diferença entre a saída predita pela rede e a saída esperada (correta), também chamda de verdade.
  + São utilizadas para penalizar a rede pela classificação incorreta de determinado vetor de entrada.
  + Muito usadas em redes de aprendizagem não supervisionada.
* Métodos de otimização
  + Treinar uma rede significa encontrar o melhor conjunto de valores para o vetor de parâmetros do modelo.
  + O algoritmo de Gradiente Descendente é um algoritmo de otimização básico.
  + Outros algoritmos indicados para Deep Learning são:
    - Matriz Jacobiana (Otimização de primeira ordem)
    - Matriz Hessiana (Otimização de segunda ordem)
* Hiperparâmetros: São as configurações que o desenvolvedor pode definir influenciando na performance da rede neural. Alguns hiperparâmetros:
  + Tamanho da camada;
  + Funções de ativação;
  + Magnitude;
  + Otimizações;
  + Minilotes (Mini-batch sizes);
  + Regularizações.
* Técnicas de Regularização

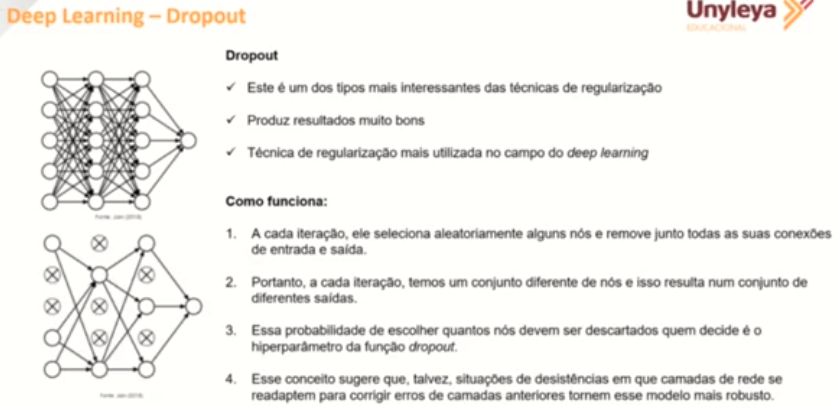




* Solucionar problema de Underfitting
  + Aumentar a quantidade de exemplos de dados de treinamento.
  + Aumentar o número de passagens nos dados de treinamento existentes.
* Solucionar problema de Overfitting
  + Uso de técnicas de regularização L2 e L1 ou Regularização Dropout (técnica mais utilizada)





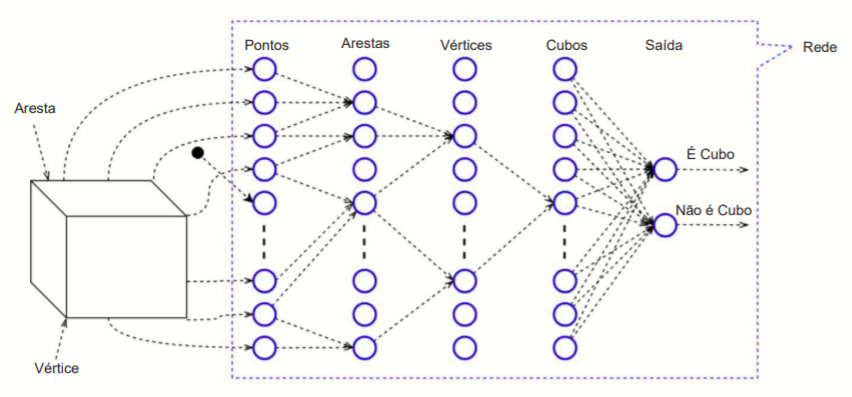


## Playground

* Como testar os conceitos de regularização e os componentes de uma rede Deep Learning?
  + Playground: <http://playground.tensorflow.org>
  + Aplicações web desenvolvida pela equipe TensorFlow

## Deep Learning - Como Funciona?

* Como funciona uma rede neural?
  + Uma rede neural não aprende apenas a reconhecer determinado objeto.
  + A rede aprende a reconhecer as características dos objetos que foi treinada.
  + Classificar estes objetos baseando-se nas suposições que fez a partir deste aprendizado
* Funcionamento
  + Aprendizado por camadas até a definição da saída



* Tipos de redes profundas
  + Multilayer Perceptrons (MLPs): é uma rede com feedforward propagation, com camadas totalmente conectadas, e pelo menos uma camada oculta.
  + Redes Neurais Convolucionais (CNNs): é uma rede neural feedforward com vários tipos de camadas especiais. Atualmente, é muito utilizada na visão computacional e no processamento de linguagem natural.
  + Redes Recorrentes: Tem uma memória interna que é baseada em porte ou em todos os dados de entrada já fornecidos à rede. É adequado para informações que funcionam de forma sequencial, como textos ou séries temporais.
  + Autoencoders: Algoritmo não supervisionado que transforma os dados brutos de entrada em características que os representam (função codificadora, chamada **encoder**), e então, voltam os dados para o formato original (função decodificadora, chamada decoder)

## Principais Frameworks

* Bibliotecas populares de código aberto: TensorFlow, Keras e PyTorch.
* TensorFlow: Desenvolvido pela Google. Pode ser configurado para utilizar processamento da GPU. É também possível definir qual CPU ou GPU utilizar para o processamento.
* Keras: Biblioteca em Python de alto nível. Roda sobre o TensorFlow, mas também possui suporte ao CNTK (Microsoft) ou Theano. É possível realizar experimentos rápidos e ele é relativamente fácil de usar em ocmparação ao TensorFlow.
* PyTorch: Desenvolvido pelo Facebook. Biblioteca de Deep Learning para Python. Relativamente fácil de usar.

## Algoritimo - Redes Neurais

* Ver Notebook: “02 - Exemplo de código de uma rede neural para a função XOR.html”

## Algoritimo de Processamento de Texto

* Ver Notebook: “03 - Processamento de texto.html”

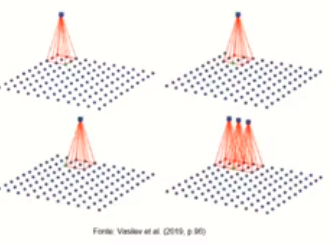
## Algoritimo de Processamento de Imagem

* Ver Notebook: “04 - Processamento de imagem.html”

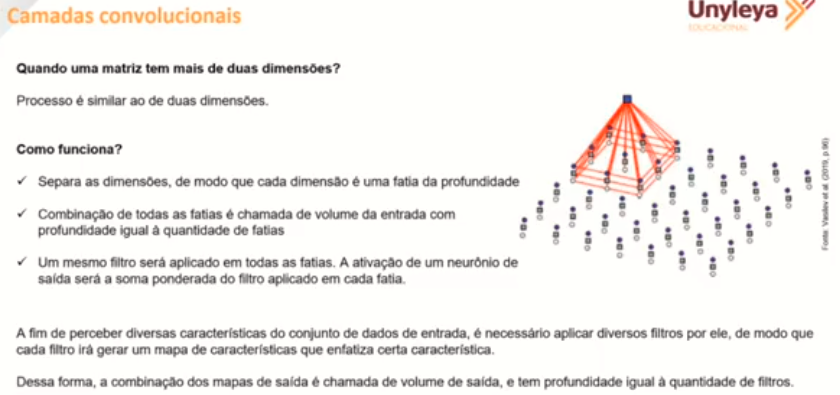
# Unidade 3 – Redes Convolucionais, Recorrentes e Adversárias

## Redes Neurais Convolucionais

* Tambpem conhecidas como Convolution Neural Networks (CNN)
* Um dos principais problemas das redes neurais simples é que elas transformam todas as entradas em dados vetoriais.
* Caso o formato seja uma matriz, pode prejudicar a análise dos dados.
* Redes Convolucionais são um tipo de rede neural para o processamento de dados organizados em forma de grade, ou matriz, como por exemplo dados de **séries temporais e imagens.**
* Vantagens
  + São muito eficientes para aplicações práticas, e receberam o nome de CNN por aplicarem uma operação matemática chamada convolução, que é um tipo de operação linear específica, nos dados que estão analisando.
* Argumentos da convolução
  + (1) Entrada de dados
  + (2) Kernel
  + (3) Saída, ocnhecida como mapa de características
* Camadas convolucionais
  + São formadas por um conjunto de filtros (kernel) e todos os filtros são aplicados em toda a extensão dos dados de entrada.
  + Um kernel é definido por um conjunto de pesos que podem ser aprendidos.
* Exemplo: Identificação e classificação de carros em imagens
  + Cada neurônio da rede neural representa a intensidade de cor de um pixel.
  + O filtro (kernel) é 3x3 e é aplicado no primeiro conjunto de pixels da imagem.
  + Cada conjunto de 9 neurônios é associado a um dos pesos do kernel. O peso é a soma ponderada dos valores dos 9 neurônios (pixels).
  + O filtro percorre toda a matriz de entrada, e a soma ponderada será calculada para cada novo conjunto de entrada.
  + O conjunto de pesos será utilizado para calcular o neurônio de ativação de toda a matriz.
  + O compartilhamento de parâmetros é feito para reduzir a quantidade de pesos utilizados e para destacar características importantes em partes específicas da matriz de entrada.
  + O processo deve ser repetido para todas as células da matriz de entrada.
  + Ao final, será gerado um conjunto de neurônios organizados espacialmente.



* Redes convolucionais para matrizes de mais de duas dimensões

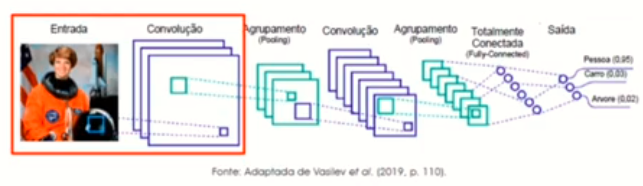


## Algoritmo – Operação Convolucional

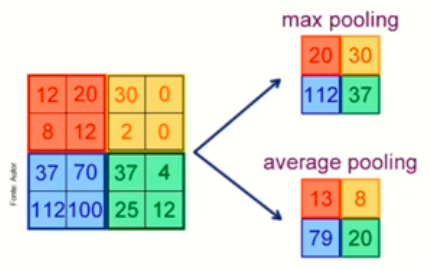
* Ver Notebook: “05 - Operação convolucional.html”

## Arquitetura CNN

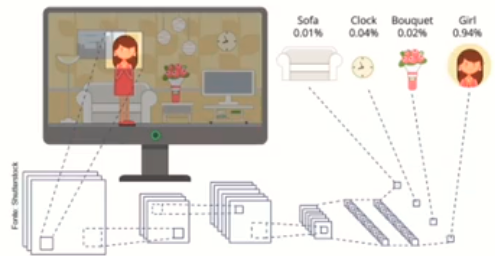
* As Redes Neurais Convolucionais possuem algumas características básicas
  + Geralmente são alternadas uma ou mais camadas de convolução com uma ou mais camadas de agrupamento.
  + À medida que a rede vai ficando mais profunda, os campos receptivos vão aumentando de tamanho.



* Camadas de agrupamento (Pooling): divide a fatia de entrada em uma grade que cada célula representa um campo receptivo de alguns neurônios, semelhantemente à camada de convolução.
  + Ao gerar essa grade, uma operação de agrupamento é aplicada em cada uma das células.
  + Não há pesos nas camadas de agrupamento
* Tipos de camadas de agrupamento
  + Max Pooling: Essa operação pega o neurônio com valor de ativação mais algo em cada campo receptivo, ou seja, em cada célula da grade, e propaga apenas este valor. Ao realizar o passo para trás, o gradiente é direcionado apenas ao neurônio que possui maior valor de ativação durante o passo para frente, os outros neurônios propagam zero.
  + Average Pooling: A saída de cada campo receptivo é a média de todos os valores de ativação do campo.



* Arquitetura de uma CNN
  + As camadas convolucionais são utilizadas para extrair características dos dados de entrada.
  + Quanto mais profunda for a camada, mais abstrata é a característica que ela consegue extrair.
  + Quanto mais profundas forem as camadas convolucionais, mais filtros ela utiliza.



## Algoritmo – Classificação de Dígitos

* Ver Notebook: “06 - Classificação de dígitos escritos manualmente.html”
* Dicas de otimizadores, ver na apostila.
* Foi possível observar que a rede neural convolucional apresentou melhor acurácia do que uma rede neural simples (estudada na Unidade 02), mesmo com menos rodadas de treinamento (epochs).
  + Algoritmo melhor otimizado: Melhores resultados em menos tempo de treinamento.

## Regularização

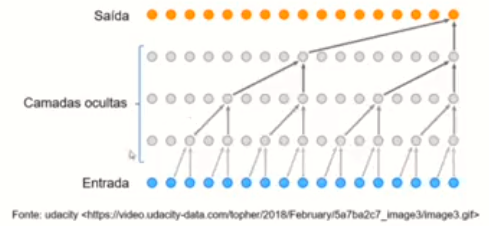
* Conceito: Qualquer tipo de modificação feita em um algoritmo de aprendizado que culmine na redução do erro de generalização (por exemplo, confundir gato com cachorro), mas não no erro de treinamento.
* Tipos de regularização:
  + Dropout
  + Aumento de dados (Overfitting)
* Dropout: Técnica que produz bons resultados. O propósito dessa técnica é fazer com que os neurônios sejam mais independentes ao aprender as características, deixando de depender tanto no que os demais neurônios aprenderam. Funcionamento:
  + A cata iteração, ele seleciona aleatoriamente alguns nós e remove junto todas as suas conexões de entrada e saída.
  + A cada iteração, temos um conjunto diferente de nós e isso resulta num conjunto de diferentes saídas, pois outras características são coletadas, melhorando a utilização da rede e dos resultados.
  + Essa probabilidade de escolher quantos nós devem ser descartados quem decide é o hiperparâmetro da função dropout.
* Aumento de dados: Quando o conjunto de treinamento é muito pequeno, a rede pode ficar muito especializada apenas neste conjunto, é o que chamamos de **overfitting**.
  + A técnica de aumento de dados é uma das técnicas de regularização mais eficientes, pois ela busca resolver o problema de overfitting aumentando o conjunto de testes artificialmente.
  + Algumas formas de aumentar a quantidade de dados de treinamento são: rotação das imagens, inversão horizontal ou vertical, aumento ou redução do tamanho, corte, inclinação, ajuste de constraste ou brilho.
* Normalização em lotes (mini-batches)
  + Permite a aplicação de um processamento de dados, semelhante à pontuação-padrão, nas camadas ocultas da rede.
  + Essa técnica normaliza as saídas de cada camada oculta em pequenos lotes (minibatch), de modo que mantém seu valor médio de ativação próximo de zero e seu desvio-padrão próximo de um.

## Algoritmo – Classificação de Imagens

* Ver Notebook: “07 - Classificar imagens de objetos.html”
  + “data\_generator = ImageDataGenerator” : Utilizado para aumentar a massa de dados.
* Comparado à rede neural de classificação de imagens da Unidade 02, na abordagem com CNN a acurácia aumentou para a mesma quantidade de ciclos de treinamento.

## Redes Neurais Recorrentes

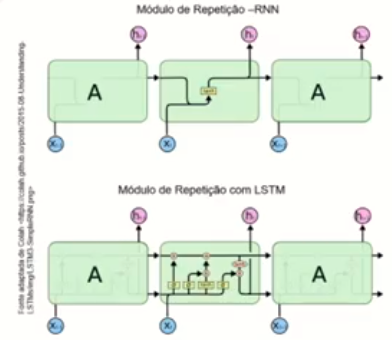
* Recurrent Neural Networks (RNN)
* São redes neurais específicas para processar dados sequenciais de tamanho variável.
* São redes em loops, que permitem que as informações persistam. Elas aplicam a mesma função sobre uma sequência de valores.
* São redes que possuem dependência temporal dos dados anteriores para calcular as saídas dos dados novos.



* RNN possuem funções de ativação não lineares.
* RNN não tem limite para dados de entrada e saída
* Possibilidades de processamento de dados de entrada e resultados de saída:
  + Um para Um: Exemplo – Uma legenda para uma imagem
  + Um para Muitos: Exemplo – Descrição textual de uma imagem
  + Muitos para Um: Exemplo – Sentimento a partir de um texto
  + Muitos para Muitos (Indiretamente): Exemplo – Tradução de texto
  + Muitos para Muitos (Diretamente): Exemplo – Reconhecimento de fala para texto (cada palavra identificada gera uma palavra escrita).
* Backpropagation ao longo do tempo: Utilizado para treinar redes recursivas.
  + Utiliza os pesos para a rede neural consiga aprender a mapear corretamente as entradas para as saídas.
  + O Backpropagation em uma RNN é desdobrado ao longo do tempo, logo, após o desdobramento da rede, teremos um comportamento similar a uma rede feedforward com multi camadas. Ou seja, cada camada oculta vai ter uma rede feedforward que pode ser comparada com a interação de uma rede recursiva, porém com várias entradas para cada camada.
* RNN são muito utilizadas para redes temporais e análise de texto (dados sequenciais).

## Algoritmo – RNN

* Ver Notebook: “08 - RNN - Implementação e treinamento.html”
* Cuidar com problema de dissipação e explosão dos gradientes
  + LSTM (Long Short-Term Memory) é um tipo especial de RNN para evitar a explosão de gradiente. Ajuda na remoção da sujeira e do ruído para evitar a explosão do gradiente.



## Processamento de Texto

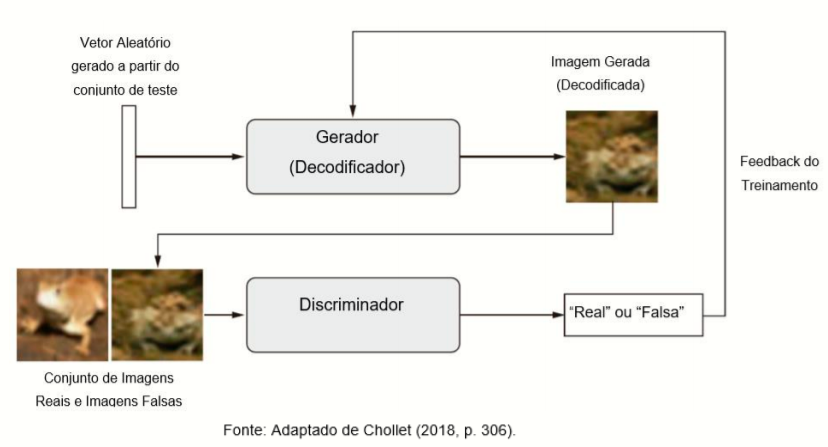
* Conceito: Modelagem de Linguagem é a tarefa de computar a probabilidade de que uma sequência de palavras faça sentido.
  + A modelagem de linguagem pode ser feita em nível de palavras ou em nível de caracteres.
* Aplicações:
  + Reconhecimento de discurso;
  + Reconhecimento de imagens de caracteres;
  + Tradução pela máquina;
  + Correção de escrita.
* Modelos
  + Define a distribuição probabilística para uma sequência de palavras.
  + Dada uma sequência de palavras de tamanho m, este modelo define a probabilidade P para toda a sequência de palavras.
* Modelo N-grams
  + Distribui a junção das palavras em várias partes independentes.
  + Então, o modelo encontra os n-grams até um valor n, e conta a ocorrência de cada n-gram naquele conjunto. A partir dessa conta, é possível estimar a probabilidade a última palavra de cada n-gram de acordo com as n – 1 palavras.
  + Exemplo de funcionamento:
    - Frase “Está fazendo sol hoje”
    - 1. gram: “Está”, “fazendo”, “sol”, “hoje”;
    - 2. gram: “Está fazendo”, “fazendo sol”, “sol hoje”;
    - 3. gram: “Está fazendo sol”, “fazendo sol hoje”;
    - 4. gram: “Está fazendo sol hoje”.
  + Maldição da dimensionalidade: A quantidade de n-grams depende do tamanho do vocabulário, que pode ser muito grande, dependendo do valor de n.
  + One-hot encoded: Ajuda o modelo a reduzir a dimensão de n-grams, criando um espaço de palavras de menor dimensão.
    - Cada palavra é codificada de forma única.
* Aprendizado: A rede aprende baseada em um texto pté-existente. É considerado **aprendizado não supervisionado.**
* Tipo de rede neural: É necessária uma rede neural simples, formada pela camada de entrada, camada oculta e camada de saída.
* Classificações do modelo com base na forma de treinamento
  + Saco contínuo de Palavras (CBOW, do inglês continuous bag of words): A rede é treinada para prever qual palavra encaixa em uma sequência de palavras onde uma palavra específica foi retirada intencionalmente.
  + Skip-gram: é o oposto do CBOW, ou seja, dada uma palavra, a rede prevê quais outras palavras a cercam.
* Modelo baseado em caracteres
  + Modelam a distribuição de acordo com uma sequência de caracteres, e não de acordo com as palavras, o que permite calcular a probabilidade em um vocabulário menor.
  + Um vocabulário, neste modelo, compreende todos os caracteres presentes no corpo do texto.
  + O lado negativo deste modelo é que, ao modelar sequências de caracteres, e não de palavras, é necessário que haja uma sequência maior para capturar a mesma informação ao longo do tempo.

## Algoritmo – Geração de Texto

* Ver Notebook: “09 - Utilizando Tensorflow para geração de novo texto.html”
* Dica de site: Kaggle

## Redes Generativas Adversárias (GAN)

* Generative Adversarial Networks
* Conceito: São redes que utilizam o aprendizado não supervisionado para treinar dois modelos em paralelo, **uma rede que constrói uma imagem**, e outra rede que verifica o quão parecida essa imagem é de uma real. Ambas as redes são treinadas imultaneamente, sempre buscando a melhoria uma da outra.
  + Uma das áreas mais avançadas que existem em Deep Learning.
* Utilização: Possui diversas aplicações práticas.
  + Aprendem sobre os objetos e criam outras versões destes objetos;
  + Geração de imagens sintéticas realistas;
  + Podem criar imagens a partir de textos;
  + Aumento de Resolução de uma imagem;
  + Desenho automático;
  + Criação de uma imagem a partir de outra imagem.
    - Projeto: <https://github.com/junyanz/IGAN>
* Funcionamento
  + A rede geradora recebe como entrada uma imagem real.
  + A imagem é analisa, dela são extraídas características.
  + Um gerador (decodificador) cria uma imagem sintética.
  + Na etapa do Discriminador a imagem sintética é comparada com outra imagem (real ou sintética). O Discriminador decide se a imagem gerada é “Real” ou “Falsa”.
  + Se a imagem gerada for considerada “Falsa”, o Discriminador realimenta o Gerador, tentando fazer a imagem sintética ficar cada vez mais parecida com a segunda imagem.
  + Logo, na rede discriminadora, ou adversária, há como entrada uma imagem (real ou sintética) e ela analisa se a imagem foi gerada pela rede geradora, ou se veio do conjunto de treinamento (imagem real).
  + Este tipo de abordagem é chamado de **redes adversárias**, pois elas são treinadas para enganar uma a outra, ou seja, a rede geradora tenta criar uma imagem tão realística que a rede discriminadora entenderá como real, e a rede discriminadora é treinada para distinguir imagens reais das sintéticas.



* Ver Notebook: “10 - Gerando novas imagens MNIST com GANs e keras.html”
* Curiosidades
  + São sistemas dinâmicos, pois tendem otimizar o modelo para que a imagem sintetizada não seja reconhecida pelo discriminador;
  + Difícil treinamento, não há um número exato de ciclos de treinamento;
  + Redes geradoras aprendem a representar as características presentes no treinamento;
  + Rede discriminadora é uma rede convolucional;
  + Utilização de Backpropagation durante o treinamento para atualizar os parâmetros da rede geradora.
* Tipos especiais de GAN:
  + DCGAN (Deep Convolutional Generative Adversarial Networks): Desenvolvida por Radford, Metz e Chintala (2015), tem uma boa performance na geração de imagens. Exemplos, códigos e maiores detalhes podem ser verificados no repositório dos autores.
    - Disponível em: <https://github.com/Newmu/dcgan_code>.
  + GAN Condicional (Conditional GAN): Apresentado por Mirza e Osindero (2014), esse tipo de GAN se utiliza, também, dos rótulos das classes, a fim de gerar imagens de uma classe específica.
  + LAPGAN: Apresentado por Denton et al. (2015), mostra que uma série de GANs condicionais podem ser treinadas para gerar uma imagem inicial de baixa resolução, e então, adicionar detalhes a essa imagem de forma incremental.
    - Aumenta a resolução da imagem.
    - O nível de detalhes nas imagens geradas por essa rede chega a confundir não apenas as redes discriminadoras, mas, também, o ser humano.
* Características
  + Permitem a geração, principalmente, de imagens sintéticas realistas, pois forçam as imagens geradas a serem estatisticamente parecidas com as imagens reais.
  + São redes que utilizam o **aprendizado não supervisionado** para treinar dois modelos em paralelo, uma rede que constrói uma imagem, e outra rede que verifica o quão parecida essa imagem é de uma real.
  + Ambas as redes são treinadas simultaneamente, sempre buscando a melhoria uma da outra.
  + As GANs são sistemas dinâmicos, nos quais a otimização procura sempre o equilíbrio entre as imagens sintetizadas e as reais, de modo que o discriminador não consiga distinguir entre elas.
  + Em uma GAN, a rede discriminadora é uma rede convolucional típica, o que permite o treinamento paralelo dela e da rede geradora. Já a rede geradora utiliza uma camada especial, chamada de camada deconvolucional (deconvolutional layer).

## Unidade 4 - Considerações Finais

## Transferência de Aprendizado

* Conceito: É o processo de aplicar um modelo já treinado a um novo problema que seja correlacionado a este modelo.
  + É possível pegar uma rede já treinada utilizando a base de dados ImageNet e reaproveitá-la para a classificação de um tipo específico de imagens, como imagens de animais.
* ImageNet
  + Link: <http://image-net.org/>
  + É um grande banco de dados visual projetado para uso em pesquisa de software de reconhecimento de objetos visuais.
  + Está integrado com TensorFlow, Keras e PyTorch
    - Todas já possuem arquiteturas pré-treinadas na base de dados ImageNet.
* A transferência de aprendizado pode ser entendida como a tradução para outra linguagem desejada.
  + Começamos com as características da rede, ou seja, a saída da última camada convolucional ou de agrupamento.
  + Então, traduzimos estes valores a um novo conjunto de classe, relativas à nova tarefa.
  + Isso pode ser feito ao remover a última camada totalmente conectada de uma rede pré-treinada e substituindo-a por outra camada, que representa as classes do novo problema.
  + Porém, não basta simplesmente substituir a última camada por outra, é necessário que o novo modelo adaptado seja treinado com dados referentes à nova tarefa.
* Treinamento
  + É possível utilizar parte da rede original
  + Também é possível afinar o treinamento da rede inteira

## Algoritimo - Transferência de Aprendizado

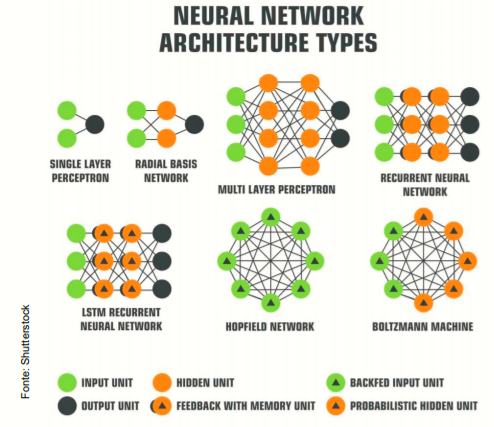
* Ver Notebook: “11 - Exemplo de transferência de aprendizado utilizando PyTorch.html”

## Revisão - Deep Learning

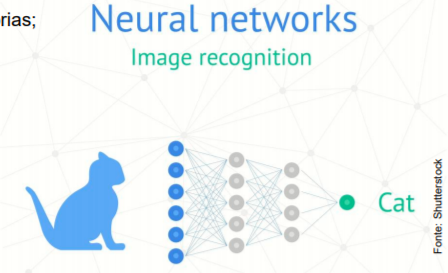
* Deep Learning
  + Deep Learning é uma revolução que está progredindo rapidamente.
  + Não é sinônimo de Inteligência Artificial ou de Aprendizado de Máquina;
  + É uma das várias abordagens de ML;
  + Utilização de GPU;
  + Tudo pode ser considerado como um ponto em um espaço geométrico. Transforma os dados em vetores, e em espaços geométricos, e então, incrementalmente aprende as transformações geométricas complexas que mapeiam um espaço (entrada) de acordo com o outro (alvos).
* Passos para desenvolver uma boa arquitetura de Aprendizado de Máquina/Deep Learning
  + **Definir o Problema:** Quais dados estão disponíveis, o que você está tentando descobrir nestes dados.
  + **Definir uma forma de verificar, medir, sua taxa de sucesso:** Medir a acurácia de seu modelo de predição.
  + **Preparar o processo de validação para seu modelo:** Definir um conjunto de treinamento, um conjunto de validação e um conjunto de teste, de modo que os rótulos dos dados dos conjuntos de validação e teste sejam desconhecidos ao conjunto de treinamento, ou seja, as etapas de validação e teste devem ser posteriores ao treinamento.
  + **Vetorizar os dados e pré-processá-los**, por normalização, por exemplo, de modo a torná-los mais adequados para a rede.
  + **Desenvolver um modelo** que demonstre que o aprendizado de máquina poderá ser utilizado para solucionar seu problema.
  + **Refine a arquitetura de seu modelo** ajustando os parâmetros e regularização. Faça mudanças baseadas na performance durante a fase de validação.
  + **Cuidado com o exagero de ajustes em seu modelo** (overfitting), para que seu modelo não seja especializado apenas nos dados do conjunto de validação

## Revisão - Deep Learning - Arquiteturas e Aplicações

* Principais arquiteturas de rede
  + Feedforward / Multilayer Perceptron;
  + Redes Convolucionais - CNN (convolutional neural networks);
  + Redes Recorrentes - RNN (recurrent neural networks);
  + Redes generativas adversárias - GANs (Generative Adversarial Networks)



* Tipos de dados vs. Tipos de Redes Neurais
  + **Dados Vetoriais:** redes densamente conectadas (camadas densas);
  + **Imagens:** redes convolucionais 2D;
  + **Som:** redes convolucionais 1D (preferencialmente) ou redes recorrentes;
  + **Texto:** redes convolucionais 1D (preferencialmente) ou redes recorrentes;
  + **Dados de séries temporais:** redes recorrentes (preferencialmente) ou redes convolucionais 1D;
  + **Outros tipos de dados sequenciais:** redes recorrentes ou redes convolucionais 1D. Preferencialmente RNN se a sequência dos dados for uma questão relevante;
  + **Vídeo:** redes convolucionais 3D (se há necessidade de capturar efeitos de movimento) ou uma combinação de ConvNet 2D para cada frame, para extração de características, com redes recorrentes ou ConvNet 1D para processar a sequência resultante;
  + **Dados volumétricos:** redes convolucionais 3D.
* Aplicações de ML/DL - Entrada de imagem e saída vetorial: Mapeamento de imagens para dados vetoriais.
  + **Assistentes de saúde:** Mapeamento de imagens médicas para predizer a presença de tumores;
  + **Veículos autônomos:** Mapear o ambiente que o carro está inserido e comandar a direção;
  + **Jogos:** Mapear tabuleiros e predizer jogadas;
  + **Ajudante de dietas:** Mapear imagens de comida e predizer a quantidade de calorias;
  + **Predição de imagens:** Mapear imagens e prever a idade da pessoa.



* Aplicações de ML /DL - Entrada dados temporais e saída vetoriais: Mapeamento de dados temporais para dados vetoriais.
  + **Previsão do tempo:** Mapear informações climáticas de uma região e predizer o tempo nos próximos dias.
  + **Interfaciar comandos do cérebro para o computador:** Mapear séries temporais do magnetoencefalograma (MEG) em comandos para o computador.
  + **Segmentação do comportamento:** Mapear a sequência de interações de uma pessoa em um website e prever a probabilidade de ela comprar algo.
* Aplicações de ML /DL - Entrada texto e saída texto: Mapear texto para texto.
  + **Resposta Inteligente:** Mapear e-mails e sugerir respostas rápidas;
  + **Responder a perguntas:** Mapear respostas a questões de conhecimentos gerais;
  + **Resumir:** Mapear longos textos em textos mais curtos.
* Aplicações de ML /DL - Entrada texto e saída imagem: Mapear texto para imagens.
  + **Geração de imagem:** Mapear uma pequena descrição para imagens que a descrevam;
  + **Geração ou seleção de logomarca:** Mapear o nome e descrição de uma empresa e criar uma logomarca.
* Aplicações de ML /DL - Entrada texto e saída imagem: Mapear texto para imagens.
  + **Geração de imagem:** Mapear uma pequena descrição para imagens que a descrevam;
  + **Geração ou seleção de logomarca:** Mapear o nome e descrição de uma empresa e criar uma logomarca.
* Aplicações de ML /DL - Entrada imagem + texto e saída texto: Mapear imagem e texto para texto.
  + **Responder a Perguntas:** Mapear imagens e questões sobre o conteúdo da imagem para respostas textuais.
* Limitações do Deep Learning: Aplicações que requerem raciocínio ou que requerem abstração de conhecimento a partir de situações não vivenciadas.

## Algoritimo de Classificação de Cães e Gatos

* Ver Notebook: “12 - Atividade Extra - Reconhecendo Cães e Gatos.html”

## Algoritimo de Classificação de Cães e Gatos – Resposta

* Ver Notebook: “12.1 - Atividade Extra - Reconhecendo Cães e Gatos (Resposta)”

## IA, ML, DL e Ciência de Dados

* Técnicas de IA, ML e DP podem auxiliar na Ciência de Dados na análise de dados econômicos, financeiros e sociais, na detecção de padrões e insights para tomadas de decisão.
* Plataformas de competição
  + Kaggle
  + Driven Data

## Aplicando o Conhecimento – Exemplos de Aplicações

* IA aprendendor a jogar
  + Objetivo: Treinar seu algoritmo para que aprenda a jogar sozinho.
* Crie seu jogo
  + Objetivo: Interação entre IA e pessoas.
* Mercado Financeiro
  + Objetivo: Monitoramento do Mercado Financeiro
* Cidades Inteligentes
  + Objetivo: Reconhecimento facial
* Reconhecimento de carros e pessoas
  + Objetivo: Identificar movimento, identificar pessoas, objetos.
* Reconhecimento de placas de veículos
* Reconhecimento de situações de risco