# Levando IA para Produção – Notas de Aula

# Unidade 1 - Inteligência artificial - Machine Learning e Deep Learning e como publicar em produção

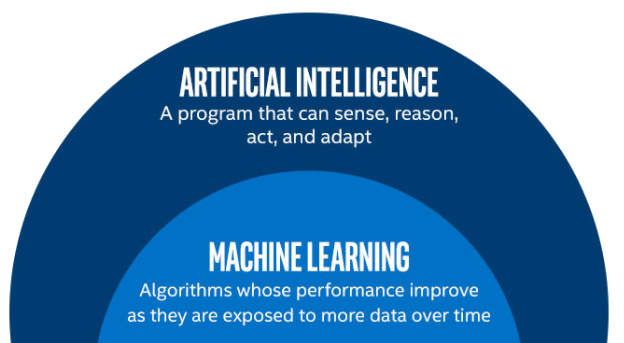
* **Objetivos:** 
  + Apresentar os conceitos de inteligência artificial, Machine Learning e Deep Learning;
  + Demonstrar as diferenças e características dos três conceitos apresentados;
  + Descrever os algoritmos de aprendizado de máquina;
  + Detalhar alguns algoritmos de Deep Learning;
  + Abordar o Big Data.

## Conceitos sobre Inteligência Artificial

* **Utilização de IA:** área de jogos, aplicações empresariais, robótica, sistemas de segurança, saúde, entre outras.
* **Técnicas de IA:** redes neurais, algoritmos genéticos, árvores de decisão, regras de associação.
  + São utilizadas em Machine Learning e Mineração de dados.
* **Utilização da IA:** sistemas de recomendação de músicas e filmes (Netflix e Spotify), dentro dos buscadores web, no funcionamento das redes sociais, etc.

## Aprendizado de Máquina (Machine Learning)

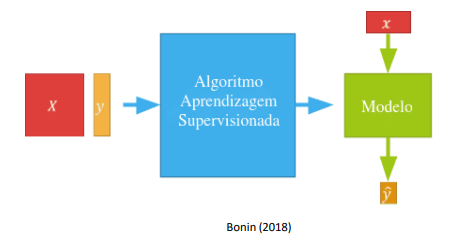
* Aprendizado de máquina é uma subárea da inteligência artificial.



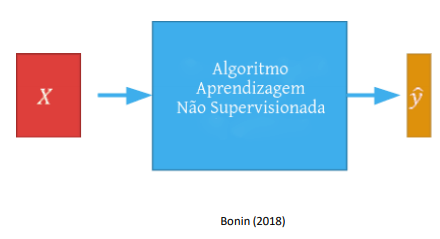
* Aprendizado de máquina é **fazer máquinas aprenderem**, mas para aprender **é preciso raciocinar**, que é o objetivo da inteligência artificial.
  + Toda abordagem de aprendizagem de máquina é também uma abordagem da inteligência artificial.
* **Formas de aprender:** com base em experiências passadas, realizando comparações, inferindo ou identificando relações.
* **Objetivo:** Aprender para auxiliar na tomada de decisão.
* **Tipos de aprendizado**
  + **Aprendizado Supervisionado**
  + **Aprendizado Não Supervisionado**
  + **Aprendizado por Reforço**

****

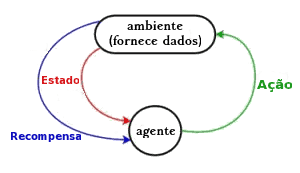
* **Aprendizado Supervisionado:** Necessita de base de dados para alimentar o algoritmo. Fases de treinamento e teste.
  + **Treinamento:** Uso de dados de entrada para análise e dados de saída conferência dos resultados da análise. Assim, o algoritmo consegue saber seu nível de assertividade.
  + **Teste:** Uso de dados de entrada desconhecidos, sem dados de saída para conferência. O algoritmo deve ser validado a partir dos resultados gerados.



* **Aprendizado Não Supervisionado:** Algoritmo deve aprender com os dados de entrada. Não possui base de dados de treinamento.



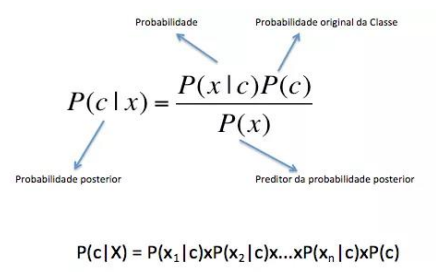
* **Aprendizado por Reforço:** Método por tentativa e erro com recompensa para os acertos. É uma abordagem semelhante a um bebê aprendendo coisas novas. A máquina não conhece nada no começo e tenta cosias novas. Para bons resultados há uma recompensa e a partir dela as futuras decisões são modificadas. Após inúmeras tentativas a máquina aprender e torna-se melhor. Eventualmente irá apenas tomar boas decisões.

****

* **Uso de Aprendizado de Máquina**
  + Campanhas de marketing cada vez mais específica e direcionada dentro da internet;
  + Sistemas utilizados por empresas financeiras para detectar fraudes;
  + Aplicações de comércio eletrônico para detectar fraudes;
  + Otimização do estoque, entendendo o comportamento de consumo dos clientes;
  + Outras.
* **Algoritmos de aprendizado de máquina**
  + Árvores de decisão;
  + Classificador de Naive Bayes;
  + Método dos mínimos quadrados;
  + Regressão logística;
  + Suporte de máquinas de vetores;
  + Decomposição em valores singulares;
  + Redes neurais.

## Algoritmo Naive Bayes

* É um classificador construído com base no Teorema de Thomas Bayes.
* **Características:** Desconsidera a relação entre os diferentes atributos de um padrão de entrada, porque trabalha cada padrão individualmente.
  + É eficiente pois precisa de poucos cálculos para chegar no resultado.
* **Exemplo:** Queremos ver a eficácia de um classificador que diagnostica uma nova doença. Foi utilizado dados de 100 pessoas.
  + 20% das pessoas tinha a doença e 80% não tinha.
  + Das pessoas com a doença, 90% apresentaram teste positivo.
  + Das pessoas sem a doença, 30% apresentaram teste positivo.



* + **Pergunta:** Se uma pessoa recebeu resultado positivo, qual a probabilidade de ela ter a doença?
  + **Solução:**
    - **P(doença|positivo) = 20% \* 90% = 0,18**
    - **P(não doença|positivo) = 80% \* 30% = 0,24**
    - **P(doença|positivo) normalizado = 0,18 / (0,18 + 0,24) = 0,4285**
    - **P(não doença|positivo) normalizado = 0,24 / (0,18 + 0,24) = 0,5714**

## Regressão Logística

* **Descrição:** Análogo ao algoritmo da regressão linear, mas pode trabalhar com saída em valores numéricos, ao invés de classes.
  + A relação entre atributos e uma variável é dada através de uma função logística.
* **Tipos**
  + **Regressão logística binomial:** Classificação em dois grupos ou categorias. É quase um processo de determinar o que é e o que não é. Se o e-mail é spam ou não, se a célula é cancerígena ou não.
  + **Regressão logística ordinal:** Classificação com três ou mais categorias. As categorias são ordenadas e a ordem é conhecida. Por exemplo, o aluno foi ruim, regular ou ótimo.
  + **Regressão logística multinomial:** Classificação em três ou mais categorias que não possuem ordem entre si. Por exemplo, esse alimento é uma verdura, fruta ou legume.
* **Uso da regressão logística:** Principalaplicação é a filtragem de spam de e-mails.

## KNN (K-Nearest Neighbor)

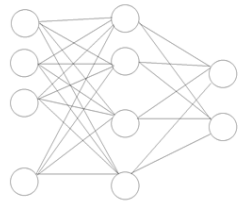
* **Descrição:** Algoritmo simples para classificação de dados. Trabalha com conceito de vizinhança. Pode ser utilizado para regressão ou classificação com as devidas adaptações.
* **Funcionamento:** K indica a quantidade de vizinhos utilizados para definir a classe da amostra.
  + K pequeno introduz ruído na classificação.
  + K grande seleciona representantes de todas as classes.
* **Etapas:** Para cada amostra
  + Calcular a distância entre a amostra e os outros elementos do conjunto;
  + Identificar os K vizinhos mais próximos;
  + Rotular a amostra com a classe mais representada por seus vizinhos.
* **Medidas de distância:** Distância euclidiana, Distância de Manhattan, Distância de Chebyshev, Outras.

## Deep Learning

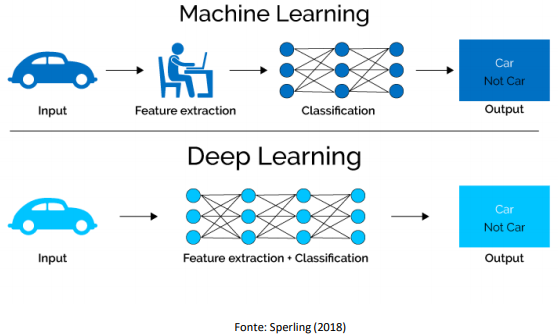
* **Descrição:** É uma subárea da inteligência artificial. Sendo mais específico ainda, é uma subárea do aprendizado de máquina. Utilização das redes neurais para reconhecimento de fala, visão computacional e processamento de linguagem natural.



* **Redes Neurais:** Buscam simular o funcionamento dos neurônios humanos e as sinapses que realizam. São construídos modelos e os neurônios são ajustados com pesos até que o modelo apresente resultados satisfatórios.
* **Estrutura básica:** Camada de entrada, camada escondida (processamento) e camada de saída (resultado).



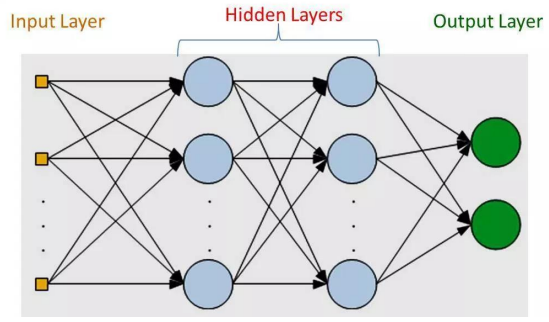
* **Funcionamento do Deep Learning:** Redes neurais com diversos camadas escondidas. Os dados de entrada são preparados para ficar em uma estrutura que propicie o aprendizado.



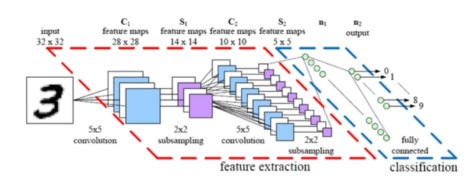
* **Usos de Deep Learning**
  + Estudo do comportamento de clientes para potencializar vendas (Ebay, Amazon), com dados de compras e navegação.
  + Reconhecimento facial em fotografias para sugestão de marcação (Facebook).
  + Processamento de linguagem natural para atendimento virtual com chatbots.

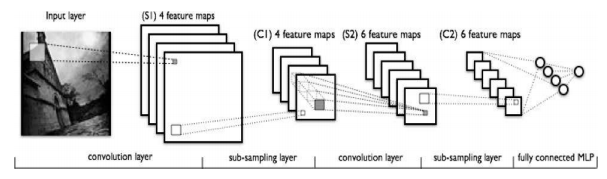
## Algoritmos do Deep Learning

* Baseados em redes neurais.
* **Multilayer Perceptron (MLP)**: Simples. Realiza classificação binária (neurônio emite saída 0 ou 1). Utiliza aprendizado supervisionado, logo é treinado com dados de entrada e saída.
  + Para classificar uma grande quantidade de classes é fundamental combinar mais neurônios na camada de saída.
  + As redes MLP podem trabalhar com a classificação de problemas não lineares, ou seja, que possuem mais de uma classe, ao contrário do Perceptron puro.
  + Devem ter ao menos uma camada escondida, além das camadas de entrada e saída.
  + O aprimoramento dos pesos ocorre com o algoritmo backpropagation, que repassa o erro no sentido contrário da rede, distribuindo para todos os neurônios.

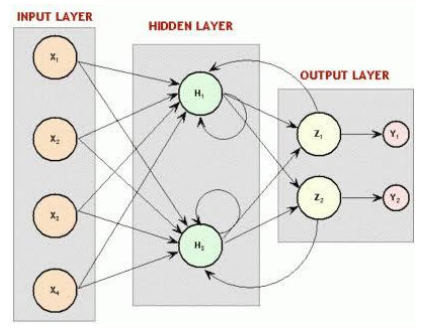


* **Redes Neurais Convolucionais (CNN):** Usadas para classificação, agrupamento por similaridade ou reconhecimento de objetos (reconhecimento de imagens).
  + Imagens são recebidas em 3 estratos (RGB), 3 matrizes.
  + A estrutura da imagem modifica entre as camadas da rede neural até classificar e reconhecer o objeto.





* Na imagem acima, da esquerda para a direita, podemos observar:
  + **Input layer**: A imagem de entrada real que é digitalizada.
  + O filtro que passa sobre a imagem de entrada é o retângulo da luz.
  + **SI:** Os mapas de ativação estão dispostos em uma pilha em cima um do outro, um para cada filtro utilizado.
  + O retângulo maior é 1 trecho para ser amostrado.
  + **CI:** Os mapas de ativação condensados ​​através de amostragem descendente.
  + **S2:** Um novo grupo de mapas de ativação gerado passando os filtros sobre a pilha que é amostrada em primeiro lugar.
  + **C2:** A segunda amostra - que condensa o segundo grupo de mapas de ativação.
  + Uma camada totalmente conectada que designa saída com 1 rótulo por nó.
* **Redes Recorrentes:** Usadas para processamento de dados sequenciais como séries temporais e linguagem natural. Funcionam com loop de feedback, alimentando a rede com o resultado n-1 para a construção do próximo passo.
  + Aprende e se transforma ao longo do tempo.



* **Outras redes**
  + Long Short-Term Memory;
  + Redes de Hopfield;
  + Máquinas de Boltzmann;
  + Deep Belief Network;
  + Deep Auto-Encoders;
  + Deep Neural Network Capsules.

## Big Data

* **Conceito:** Consiste em mecanismos computacionais para analisar um grande volume de dados. Usa ferramentas que trabalham de forma coordenada para a coleta, análise e disponibilização de dados.
* **Os 5 Vs**
  + **Volume:** Grande quantidade de dados. Uso de bancos de dados NoSQL e armazenamento e computação em nuvem.
  + **Velocidade:** Processamento de grandes quantidades de dados em tempo hábil.
  + **Variedade:** Diferentes fontes de dados e estruturas distintas.
  + **Variabilidade:** Variação de formato e conteúdo dos dados.
  + **Vínculo:** Adotar algum padrão de organização para armazenar os dados para posterior utilização.
* **Dados estruturados:** Formato conhecido. Dados de logs, resultados de formulários, etc.
* **Dados não estruturados:** Sem estrutura padrão. Dados de diferentes fontes e de conteúdos diferentes.

# Unidade 2 - Frameworks de Inteligência Artificial

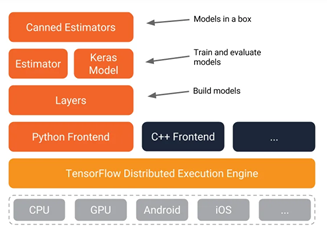
* **Objetivos:** 
  + Explicar como os frameworks e as bibliotecas apoiam a construção de aplicações de inteligência artificial.
  + Detalhar as principais bibliotecas do Python que são utilizadas em construção da IA.

## Framework

* **Características**
  + Frameworks provêm ferramentas para solução para uma família de problemas que apresentam semelhanças.
  + São aplicações “quase completas”, mas com pedaços faltando, que serão desenvolvidas para atender as características específicas de cada problema.
  + São compostos por uma grande variedade de parâmetros, garantindo que o desenvolvedor seja capaz de fazer personalizações, de acordo com as necessidades do projeto.
* **Tipos de Estruturas**
  + **Frozen spots (ou hook points):** são serviços e funcionalidades que já foram implementados pelo framework.
  + **Host spots:** são funcionalidades, serviços que devem ser implementados pelos desenvolvedores que irão inserir os seus códigos específicos à solução do problema.
* **Vantagens**
  + **Produtividade**
  + **Melhora a modularização**
  + **Aumenta a reutilização**
  + **Extensibilidade**

## A Biblioteca TensorFlow

* **Características**
  + Código aberto.
  + Desenvolvida para utilização com Python.
  + Focada em Deep Learning.
  + Oferece abstração, pois não há necessidade de implementar os algoritmos.
  + Núcleo de cálculos numéricos implementado em C++ para garantir maior velocidade de processamento.
  + Consegue trabalhar com dados em larga escala.
  + Utiliza estruturas de grafos.
  + Implementa diversos algoritmos e modelos para aprendizado de máquina e Deep Learning.
  + Possui recursos gráficos (TensorBoard).



* **História:** Criada pela Google Brain (da Google). O código open-source foi disponibilizado em 2015.
* **Usos:** Reconhecimento de textos manuscritos, reconhecimento de imagens, reconhecimentos de caracteres e processamento de linguagens naturais.
* **Vantagens**
  + Flexível, extensível e portável, podendo ser executado em qualquer dispositivo.
  + Capacidade de gerar rapidamente um produto ou serviço a partir do modelo preditivo treinado, eliminando a necessidade de reimplementar o modelo.
* **Aplicações**
  + máquina local;
  + cluster na nuvem;
  + dispositivos Android;
  + dispositivos IOS;
  + unidade de processamento personalizada Tensorflow.

## A Biblioteca Keras

* **Características**
  + Desenvolvida para Deep Learning, com recursos para a construção de redes neurais.
  + Biblioteca implementada em Python.
  + Possui redes dos tipos recorrentes e convolucionais.
  + Bastante utilizada na indústria.
  + Nível de abstração que auxilia o aprendizado para iniciantes.
* **MXNet:** Concorrente da Keras. Em constante aprimoramento. Projeto da Apache Foundation desde 2017. Código aberto e gratuito.

## A Biblioteca PyTorch

* **Características**
  + Criada para ser uma biblioteca de desenvolvimento de redes neurais (Deep Learning). Idem TensorFlow.
  + Implementada em Python, C++ e CUDA. Pensado como uma ferramenta para Python.
  + Desenvolvido pelo Facebook. Usado pelo Uber.
  + Trabalha com grafos ao longo da execução dos algoritmos.
  + Lançado em 2016.
  + Permite a construção de modelos com redes neurais complexas de forma simples.
  + Bom desempenho para processamento de cálculos numéricos.
* **Outras bibliotecas importantes:** Numpy, Scipy, Cython.
* **Principais recursos**
  + **Interface simples:** API simples de utilizar.
  + **Natureza Pythonica:** Excelente interação com Python.
  + **Diagramas computacionais:** Oferece gráficos computacionais dinâmicos.

## Comparação

**TensorFlow**

* **Características:** Biblioteca de código aberto para processamento de dados em diferentes tarefas. É uma biblioteca matemática utilizada para aplicações de aprendizado de máquina como as redes neurais.
* **Recursos:** Trabalha de modo eficiente com datasets grandes. Processamento rápido. Permite desenvolvimento em alto e baixo nível. Não é muito fácil de utilizar. Possui funcionalidade de detecção de objetos.

**PyTorch**

* **Características:** Biblioteca de aprendizado de máquina com código aberto escrita para Python. Desenvolvida pelo grupo de Inteligência Artifical do Facebook. Muito utilizada pelo meio acadêmico, pois permite manipulação direta das expressões com arrays.
* **Utilização:** Processamento de linguagem e visão computacional.
* **Recursos:** É rápida e trabalha com grandes datasets. É complexa de utilizar, mas possui facilidade para debugging. Demanda um tempo pequeno de treinamento e oferece flexibilidade para o desenvolvimento de código. Possui integração natural com Python.

**Keras**

* **Características:** Biblioteca para redes neurais de código aberto escrita em Python. Pode rodar sobre o TensorFlow. Projetada para permitir experimentação rápida com redes neurais profundas.
* **Recursos:** Boa para projetos com pequenos datasets, pois não é tão rápida quanto TensorFLow e PyTorch. Muito simples de entender e fácil de usar.

**Referências:**

* <https://www.edureka.co/blog/keras-vs-tensorflow-vs-pytorch/>
* <https://www.mygreatlearning.com/blog/computer-vision-using-pytorch/>

## A Biblioteca Numpy

* **Características**
  + Realiza operações sobre matrizes e arrays.
  + Adiciona simplicidade à realização de cálculos numéricos.
  + Possui funções para funcionalidades para manipulação de imagens.
* **ndarray: Array numpy**
* **Algumas Funções**
  + **Criação de array:** numeros = np.array([6, 7, 2, 1, 9])
  + **Dimensão do array:** numeros.shape
  + **Valores aleatórios:** numeros = np.random.random(5)

## A Biblioteca Pandas

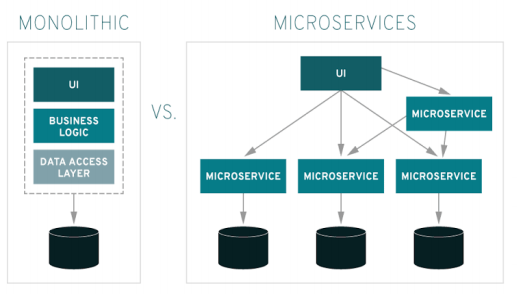
* **Características**
  + Para análise de dados, utilizada em Python
  + **Estruturas:** Series, DataFrame
    - **Series:** Como uma coluna. Cada elemento pode ser referenciado por um índice.
    - **DataFrame:** Como uma tabela. Possui diversas colunas de dados e cada linha do DataFrame pode ser referenciado por um índice
* **Leitura de dados**
  + **pd.read\_csv()**
  + **pd.read\_xlsx()**

# Unidade 3 - Microservices e como utilizar para produção

* **Objetivo:** 
  + Apresentar a arquitetura de microserviços;
  + Explicar o protocolo REST;
  + Descrever a construção de URL e URI;
  + Detalhar os arquétipos de recursos.

## Microservices

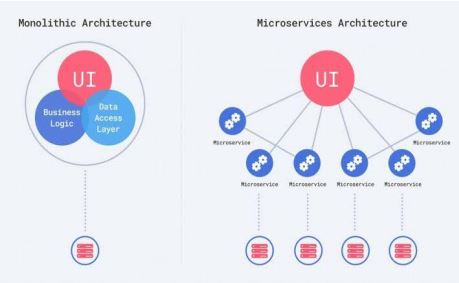
* Arquitetura utilizada para construção de softwares, oferecendo uma opção à arquitetura monolítica
  + **Arquitetura monolítica:** Estrutura única para todos os serviços do software, apenas um processo. Baixa escalabilidade.
  + **Arquitetura de microservices:** As principais funcionalidades de uma aplicação podem ser segmentadas em serviços que se complementam, mas são isolados.



* **Conceito:** Um software desenvolvido na arquitetura de microserviços é um conjunto de pequenos softwares.
  + Cada pequeno software é criado um processo específico e se comunica através de recursos muito leves.
  + A manutenção e o deploy podem ocorrer em partes específicas do software, ao invés do software completo.
* Tira vantagem do processamento paralelo.

## Quando utilizar Microservices

* **Domínios específicos:** Utilizado em domínios específicos, principalmente em domínios com baixo acoplamento entre os elementos de software.
* **Escalabilidade:** Dividir a aplicação em pequenas partes permite escalabilidade, pois cada microserviço pode ser escalado de forma independente.
  + Serviços críticos podem receber mais recursos computacionais.
* **Deploy:** Cada serviço necessita de um deploy específico. Na arquitetura monolítica o deploy é mais fácil por ser único (um processo).
* **Atualização:** Nos microserviços cada serviço pode ser atualizado individualmente, sem comprometer o funcionamento dos demais softwares.
* **Depuração:** Debbuging é mais difícil, pois é necessário avaliar o comportamento de cada serviço e os impactos que causam entre si.
* **Isolamento:** Cada microserviço pode utilizar uma tecnologia única, que não impacta no funcionamento ou desempenho dos demais serviços.



## O Protocolo REST

* **Descrição:** O REpresentational State Transfer (REST) é um protocolo utilizado para o desenvolvimento de aplicações distribuídas.
  + REST é muito utilizado no desenvolvimento de softwares na arquitetura de microserviços.
  + Muito utilizado para desenvolvimento de aplicações Web.
  + Aplicações são encaradas como recursos.
  + Cada aplicação tem um tipo e um identificador único.
* **URI:** Uniform Resource Identifier.
  + Usado para identificar cada um dos recursos da arquitetura (cada aplicação/serviço).
  + Exemplo de identificador URI: http://example.com/customers/1234
* **Princípios de funcionamento do REST**
  + Um identificador para cada elemento.
  + Criação de vínculos entre os elementos.
  + Utilização de métodos padronizados.
  + Recursos com múltiplas representações.
  + Comunicação stateless (sem estado).

## URI, URL e URN

* A URL é uma parte da URI.
  + Ex.: www.globo.com.br
* A URN (Uniform Resource Name) indica o recurso dentro da página.
  + Ex.: esportes.html
* A URI é o caminho complete do recurso.
  + Ex.: www.globo.com.br/esportes.html

## Arquétipos de Recursos

* **Tipos de recursos**
  + Document
  + Collection: Agrupamento de objetos
  + Store: Repositório gerenciado pelo cliente
  + Controller

# Unidade 4 - Produção para aplicações que utilizam inteligência artificial

* **Objetivos:**
  + Detalhar principais aplicações da inteligência artificial;
  + Apresentar o framework Flask;
  + Explicar o que é uma empresa que explora o potencial de seus dados.

## Flask

* **Características**
  + Framework escrito em Python.
  + Código aberto.
  + Permite modificações de modo simples e fácil.
  + Rapidez de desenvolvimento.
  + Arquitetura simples.
  + Permite desenvolvimento rápido.

## Ambientes de Desenvolvimento em Python

* Atom – Editor de Texto
  + Integração com Git
* Vim – Editor de Texto
  + Via linha de comando
* SciTE – Editor de Texto
  + Suporte para Python.
  + Suporte de recurso de debug.
* Pycharm – IDE
* Eclipse – IDE
  + Necessário plugin ppydev