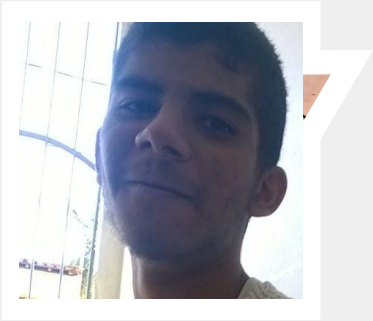
Desenvolvendo algoritmos classificadores com Python



Yuri Sales

Análise e Desenvolvimento de Sistemas - IFRN

Facebook: /yurishenrique

Twitter: @yurishenrique

Github: @yuriscosta

Ementa

1. **Introdução à linguagem Python**
    a. Variáveis e Tipos
    b. Operadores
    c. Entrada de Dados
    d. Estrutura de decisão
    e. Estruturas de repetição
    f. Funções
2. **Inteligência Artificial**
    a. O que é IA?
    b. Aplicações
3. **Machine Learning**
    a. Introdução
    b. Métodos Classificadores
    c. Biblioteca Scikit-Learn
    d. Criando algoritmos classificadores

Introdução à linguagem Python

1. **Por que usar python?**
    a. Fácil de aprender;
    b. Permite focar no problema e não na sintaxe;
    c. Todas as bibliotecas são free;
    d. Possui uma alta demanda no mercado;
    e. Possui bibliotecas robustas para trabalhar com ML.

Variáveis e Tipos

1. As variáveis não precisam ser declaradas, mas devem ser iniciadas antes de serem utilizadas;

2. O nome das variáveis são padronizados, ou seja, não pode começar com números e não pode ter espaços. Além disso, a linguagem é Case Sensitive;

3. Não possuem valor fixo, podem receber qualquer tipo de dado como números, strings, funções, classes, etc.

Variáveis e Tipos

```
1  #_*_ coding: utf-8 _*_
2
3  #certo
4  nome = "Maria"
5  print nome
6
7  #errado
8  print idade #variável idade não foi declarada
9
10 #certo
11 cidade1 = "Natal"
12 cidade2 = "São Luís"
13
14 #errado
15 1cidade = "Natal"
16 2cidade = "São Luís"
```

Variáveis e Tipos

- Valores numéricos
  - Números inteiros (int);
  - Números de ponto flutuante (float);
  - Valores booleanos (bool);
  - Complexo (complex).

```
1  #_*_ coding: utf-8 _*_
2
3  media_parcial = 60
4  media_final = 83.5
5  ligado = True
6  print type(media_parcial)
7  print type(media_final)
8  print type(ligado)
```

Variáveis e Tipos

- Strings

```
1  #_*_ coding: utf-8 _*_
2
3  str = "Instituto Federal"
4  print str #exibe a string
5  print str[0] #exibe o primeiro elemento
6  print str[4:10] #exibe entre o 4° e o 10° caractere
7  print str[10:] #exibe a partir do 10° caractere
8  print str * 4 #exibe a string 4 vezes
9  print str + " Campus Monte Castelo" #concatena e exibe a string com o novo valor
```

Variáveis e Tipos

- Listas

```
1  # coding=UTF-8
2
3  lista = ["Farofa", "Rapadura", 12345, False, 128.2]
4  nova_lista = ["Reviver", 1800]
5
6  print lista #exibe a lista inteira
7  print lista[3] #exibe o 4° elemento
8  print lista[0:2] #exibe os valores entre os índices 0 e 2
9  print lista[3:] #exibe a partir do 4° índice
10 print nova_lista * 2 #exibe 2 vezes a nova_lista
11 print lista + nova_lista #concatena e exibe as listas
```

Variáveis e Tipos

- Tuplas

```
1  # coding=UTF-8
2
3  tupla = ("Computador", "Mesa", "Fogão", True, 347)
4  nova_tupla = (12, False)
5
6  print tupla #exibe a tupla inteira
7  print tupla[3] #exibe o 4° elemento
8  print tupla[0:2] #exibe os valores entre os índices 0 e 2
9  print tupla[3:] #exibe a partir do 4° índice
10 print nova_tupla * 2 #exibe 2 vezes a nova_tupla
11 print tupla + nova_tupla #concatena e exibe as tuplas
```

Variáveis e Tipos

- Dicionários

```
1  # coding=UTF-8
2
3  dic = {"Nome": "Marcelo", "Matricula": 201610134, "Curso": "Informática"}
4
5  print dic #exibe todo o dicionario
6  print dic['Curso'] #exibe o valor que está em Curso
7  print dic['Nome'] #exibe o valor que está em Nome
8  print dic.keys() #exibe todas as chaves
9  print dic.values() #exibe todos os valores
```

Operadores Aritméticos

+	Soma
-	Subtração
*	Multiplicação
/	Divisão
//	Divisão de inteiros
**	Potenciação
%	Resto da divisão (módulo)

Operadores Lógicos

>	Maior
<	Menor
>=	Maior ou igual
<=	Menor ou igual
==	Igual
!=	Diferente
not	Negação
and	E
or	Ou

Entrada de dados

- raw_input()

```
1  # coding=UTF-8
2
3  nome = raw_input("Digite o seu nome: ")
4  idade = int(raw_input("Digite a sua idade: "))
5  passagem = float(raw_input("Quanto custa a passagem? "))
6
7  print("Seu nome é: %s" %nome)
8  print("Você tem %d anos" %idade)
9  print("A passagem custa R$ %0.2f" %passagem)
```

Estrutura de decisão

- As estruturas de decisão alteram o fluxo de execução do algoritmo baseado em testes lógicos.

```
1  # coding=UTF-8
2
3  nome = raw_input("Digite seu nome: ")
4
5  if nome != "Marcos":
6      print "Seu nome não é Marcos"
```

```
1  # coding=UTF-8
2
3  x = int(raw_input("Digite um número: "))
4  if x > 0:
5      print "x é positivo"
6  else:
7      print "x é negativo"
```

```
1  # coding=UTF-8
2
3  idade = int(raw_input("Digite sua idade: "))
4
5  if idade >= 0 and idade < 12:
6      print "Criança"
7  elif idade > 12 and idade < 18:
8      print "Adolescente"
9  elif idade >= 18 and idade < 60:
10     print "Adulto"
11 elif idade >= 60:
12     print "Idoso"
13 else:
14     print "Idade inválida!"
```

Estruturas de repetição

- While

```
1  # coding=UTF-8
2
3  i = 0
4  while i < 100:
5      print i
6      i = i + 1
```

```
1  # coding=UTF-8
2
3  i = 0
4  while i < 5:
5      print ("%d é menor que 5" %i)
6      i = i + 1
7  else:
8      print ("%d é igual a 5" %i)
```

```
1  # coding=UTF-8
2
3  i = 1
4  while True:
5      if i % 2 == 0 and i % 3 == 0:
6          print i
7          break;
8      i = i + 1
```

```
1  # coding=UTF-8
2
3  i = 0
4  while i < 100:
5      i = i + 1
6      if i % 5 == 0 and i % 10 == 0:
7          print i
8          continue;
```

Estruturas de repetição

```
1  # coding=UTF-8
2
3  cidades = []
4  print "---Selecione uma opção abaixo---"
5  i = int(raw_input("1. Cadastrar cidade\n0. Sair\n"))
6
7  while i != 0:
8      cidades.append(raw_input("Digite o nome de uma cidade: "))
9      print "---Selecione uma opção abaixo---"
10     i = int(raw_input("1. Cadastrar cidade\n0. Sair\n"))
11 else:
12     print "\nPrograma fechado com sucesso!"
13
14 if (len(cidades) == 0):
15     print "\nNenhuma cidade foi cadastrada."
16 else:
17     string_cidade = ", ".join(cidades)
18     print string_cidade
```

Estruturas de repetição

- For

```python
# coding=UTF-8

for i in range(10):
    print i

for i in range(10, 15):
    print i

for i in range(50, 100, 5):
    print i

for i in range(10, 0, -2):
    print i
```

```python
# coding=UTF-8

lista = ["banana", "uva", "maçã"]
for i in lista:
    print i

tupla = (1, 2, 3, 1000)
for i in tupla:
    print i

for i in range(51):
    if (i % 2 == 0):
        print ("%d - Par" %i)
    else:
        print ("%d - Ímpar" %i)

dicionario = [1: "Roberto", 2: "Messias", 3: "Alexandre"]
for i in dicionario:
    print i.value()

for k, v in dicionario.iteritems():
    print k, v
```

Estruturas de repetição

- For

```
1   # coding=UTF-8
2
3   for i in range(1, 10, 2):
4       if (i % 5 == 0):
5           print i
6           break
7       else:
8           print i
9           continue
10
11  for i in range(2, 10):
12      for x in range(2, i):
13          if i % x == 0:
14              print ("%d = %d * %d" %(i, x, i/x))
15              break
16      else:
17          print i, 'é um número primo'
```

Funções

```python
# coding=UTF-8

def soma(x, y):
    return x + y

def quadrado(x):
    return x**2

def soma_quadrados(x, y):
    x = quadrado(x)
    y = quadrado(y)
    return soma(x, y)

print soma_quadrados(11, 4)
```

```python
# coding=UTF-8

def fatorial(x):
    if x == 0:
        return 1
    else:
        return x * fatorial(x-1)

print fatorial(5)
```

Funções

```
1  # coding=UTF-8
2
3  def faculdade(periodo, *args, **kwargs):
4      print "Período: %d\nargs: %s\nkwargs: %s" %(periodo, args, kwargs)
5
6  faculdade(2, 6, 3, materia1 = "Física", materia2 = "Algoritmos",
   materia3 = "Web Design")
7
8  def sistema(so = "windows"):
9      if so == "windows":
10         print "Aposto que usa windows 10"
11     elif so == "linux":
12         print "Aposto que usa ubuntu"
13     else:
14         print "pra ter um mac tem que ser rico \o/"
15
16 meu_sistema = "WindowS"
17 sistema(meu_sistema.lower())
```

Desafio

Faça uma agenda de contatos, onde cada contato deve ter um nome, telefone e email. A agenda deve cadastrar, exibir, editar e remover os contatos.

Inteligência Artificial

Inteligência Artificial

- O que é inteligência?
  - A capacidade de lidar com com novas situações; a capacidade de solucionar problemas, de responder a questões, de engendrar planos e assim por diante.
- E o que é Inteligência Artificial?
  - “Inteligência Artificial é o estudo dos sistemas que agem de um modo que a um observador qualquer pareceria ser inteligente”;
  - “Inteligência Artificial envolve utilizar métodos baseados no comportamento inteligente de humanos e outros animais para solucionar problemas complexos”.

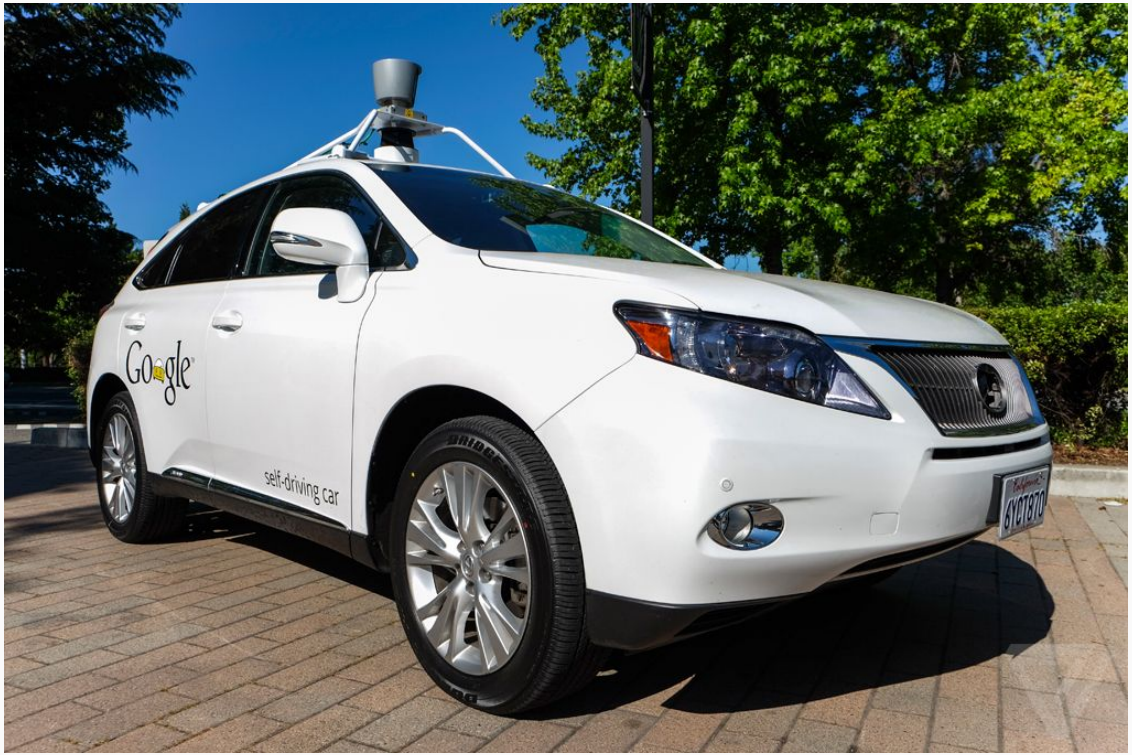
Inteligência Artificial Forte

Dando uma capacidade suficiente de processamento e fornecendo suficiente inteligência, pode-se criar uma máquina que pensa e é consciente como um ser humano.

Inteligência Artificial Fraca

É uma visão de que comportamento inteligente pode ser modelado e utilizado por computadores para solucionar problemas complexos.

O fato de um computador agir inteligentemente não significa que ele seja inteligente como um ser humano.



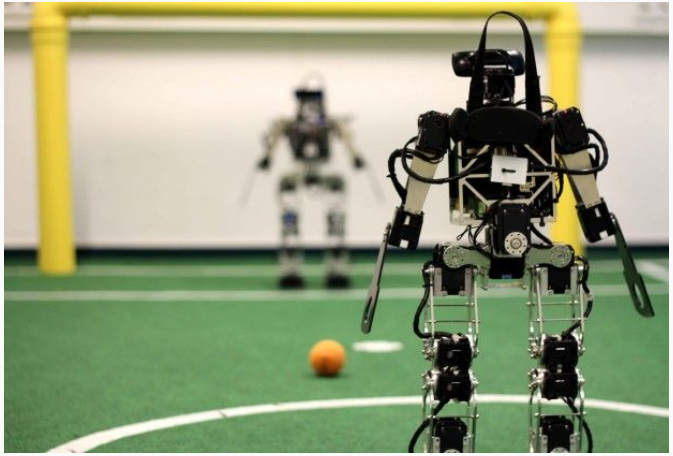
Métodos fracos x Métodos fortes

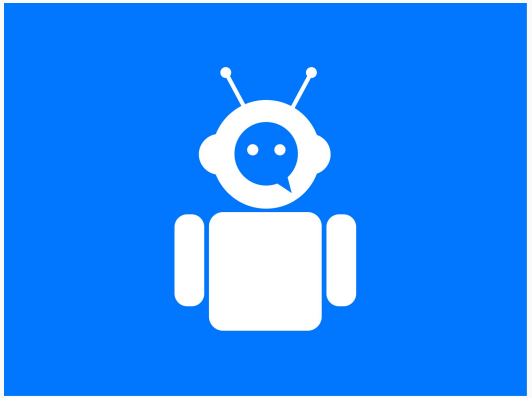
- Métodos fracos
  - Usam lógica, raciocínio automatizado e outras estruturas gerais que podem ser aplicadas a uma gama de problemas, mas que não necessariamente incorporam qualquer conhecimento profundo sobre o mundo do problema que está sendo solucionado.
- Métodos fortes
  - O sistema dispõe de sólidos conhecimentos sobre o seu mundo e quais problemas devem encontrar;
  - Depende dos métodos fracos. É inútil ter conhecimento e não ter metodologia para lidar com este conhecimento.

Aplicações


Google
self-driving car


QATAR AIRWAYS
EA SPORTS
FIFA 16
FIFA
Official Licensed Product
PEGI 3





Áreas de estudo da IA

- Sistemas baseados em Agentes e Múltiplos Agentes;
- Busca;
- Planejamento Automatizado;
- Processamento de Linguagem Natural;
- Representação do Conhecimento;
- Algoritmos Genéticos;
- Raciocínio e Raciocínio Probabilístico;
- Robótica;
- Machine Learning.

Machine Learning

Machine Learning

- Traz a Ciência da Computação e Estatística juntas para prever futuros resultados;
- Aprender a partir de exemplos;
- Cada exemplo tem um número de características e atributos para descrevê-lo;
- Se você escolher uma característica e ela te der a informação correta, então o código poderá classificar novos exemplos.
- *“Machine Learning is a subfield of computer science that gives computers the ability to learn without being explicitly programmed.”* (Arthur Samuel, 1959);
- *“A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E”* (Tom Mitchell).

Machine Learning

Ex.: Quando procuramos algo no Google, o que acontece em seguida? Ele nos mostra milhares de páginas com as as palavras-chaves relevantes. Mas o que acontece por dentro? Como ele nos mostra as páginas relevantes conforme as nossas buscas? O Google lida com milhares de trilhões de buscas por dia, como ele lida com todos esses dados com uma precisão correta?

Mostrar as páginas relevantes conforme as palavras-chaves (Tarefa T), você pode executá-la por meio de um algoritmo de Machine Learning com dados das pesquisas anteriores e seus padrões (Experiência E) e caso tenha aprendido com sucesso, ele irá, então, fazer uma predição melhor das futuras pesquisas (Performance P).

Passos utilizados em ML

- Coleta de dados: é o primeiro passo e a base de todo o conjunto de dados. Deve ser planejado com cuidado.
- Apresentação dos dados: a grande quantidade de dados coletadas devem ser apresentada de forma adequada e concisa para análises futuras. Os dados que foram coletados podem conter erros e problemas de qualidade como dados ausentes, valores abertos, erros de texto, etc..
- Treinar um modelo: o conjunto de dados “enxuto” é dividido em dois tipos: dados para treinamento e dados para teste. Os dados de treino são utilizados para desenvolver o modelo e os de teste são utilizados como uma referência.

Passos utilizados em ML

- Avaliar o modelo: para testar a precisão, o modelo de teste é utilizado. Esse passo determina a precisão na escolha do algoritmo baseado no resultado.
- Melhorar a performance: esse passo envolve melhorar a precisão. Escolhendo modelos diferentes pode melhorar a eficiência. Se gastarmos um tempo na coleta de dados e na preparação, podemos ter uma precisão melhor.