Aprendendo Machine Learning com Python

Workshop de ciência de dados

Giana de Almeida

Aprendizado de máquina

Bom, nós já descobrimos que os computadores precisam ser ensinados.

Mas podemos pensar em uma série de aplicações que tratar disso de forma por instruções pode se tornar inviável

Fefa

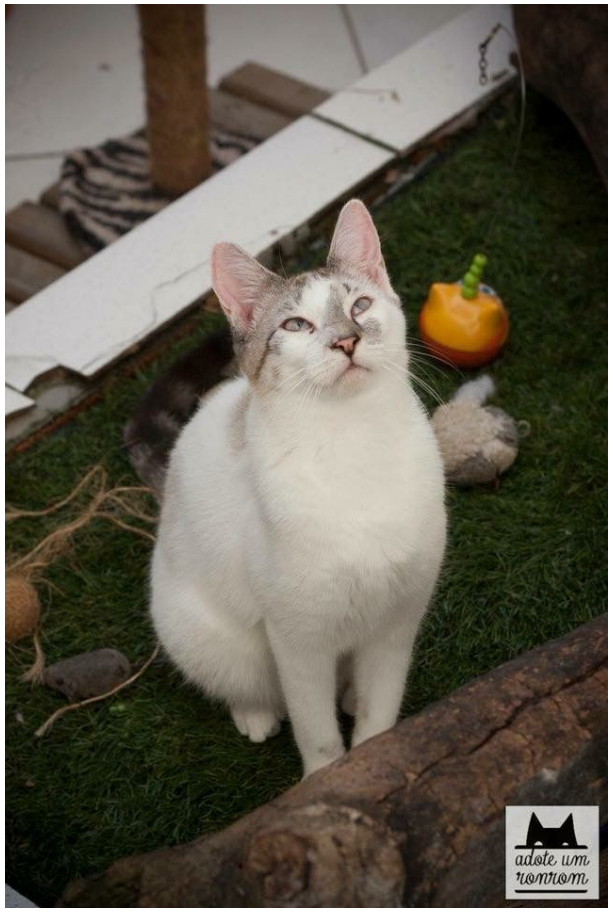
E agora?

Como resolver isso?

Como resolver isso?

A proposta é ensinar o computador a aprender por meio das próprias experiências.

Kurt



Como ensinar ??

A primeira vez que você viu um gato, provavelmente você não gravou que deveria chamá-lo (~~classificá-lo~~) como **gato**

Mimi

Como ensinar??

Mas a medida que você recebeu essa informação, você foi associando “gato” com o animal

Lia

Como ensinar??

E a diferenciar de outros animais

Hmm... E daí?

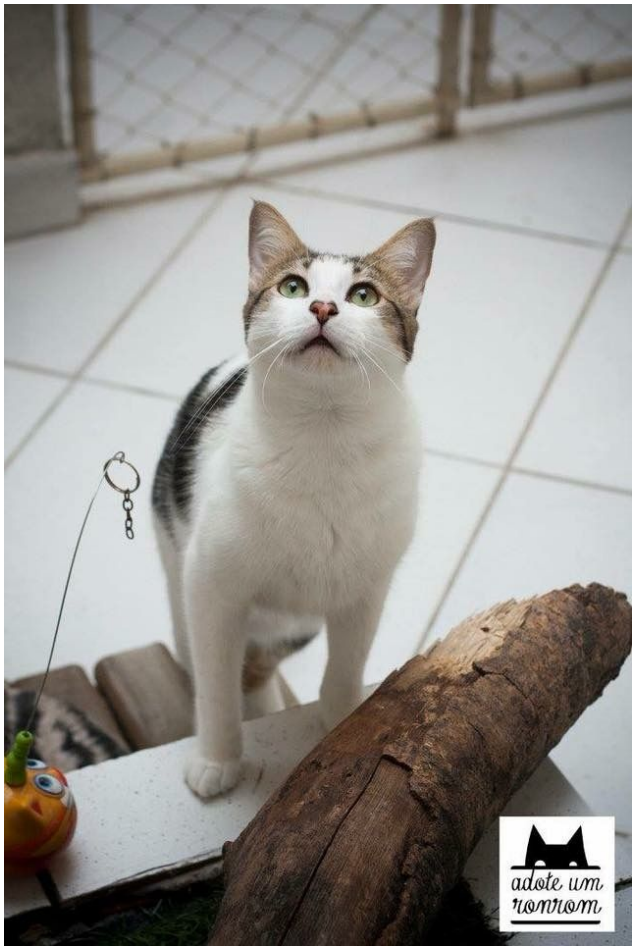
Com o computador faremos parecido.

Lúcio

Aprendizagem de máquinas

A ideia é que ao invés de você programar uma “solução” pronta e pensar em todas as situações possíveis…

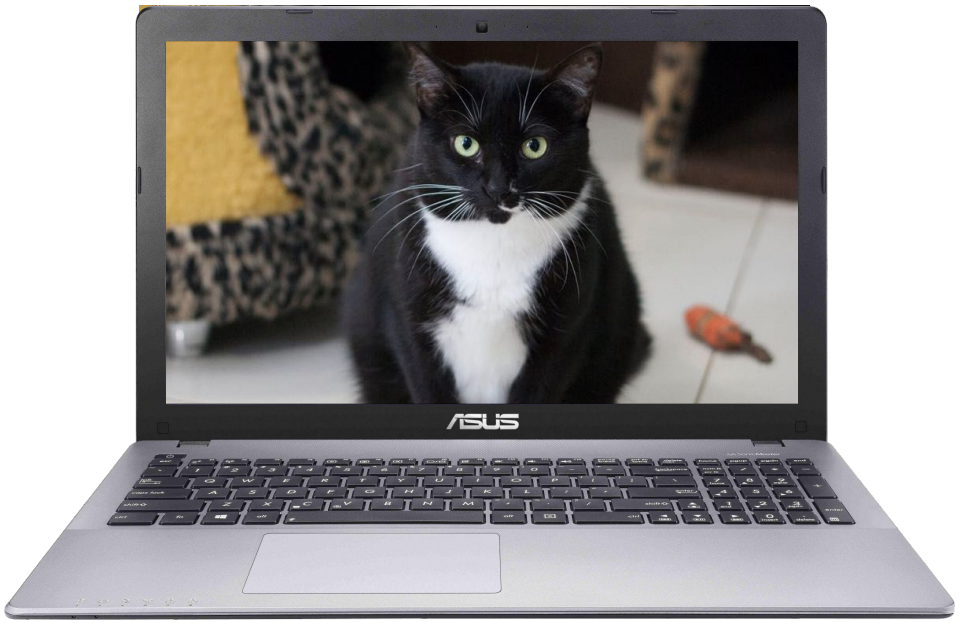
Artur



Aprendizagem de máquinas

Que o programa consiga analisar, se auto gerenciar e fazer isso por conta

Viola



Aprendizagem de máquinas

Conseguimos verificar que um computador aprende por meio de uma experiência(amostra de dados) com uma tarefa a resolver, e que isso possa ter sua performance medida, sendo essa melhor que o modo anterior de realizar essas tarefas

Furiosa

Não tô entendendo nada...

Aprendizagem de máquinas

Vamos dividir os conceitos e exemplificar então:

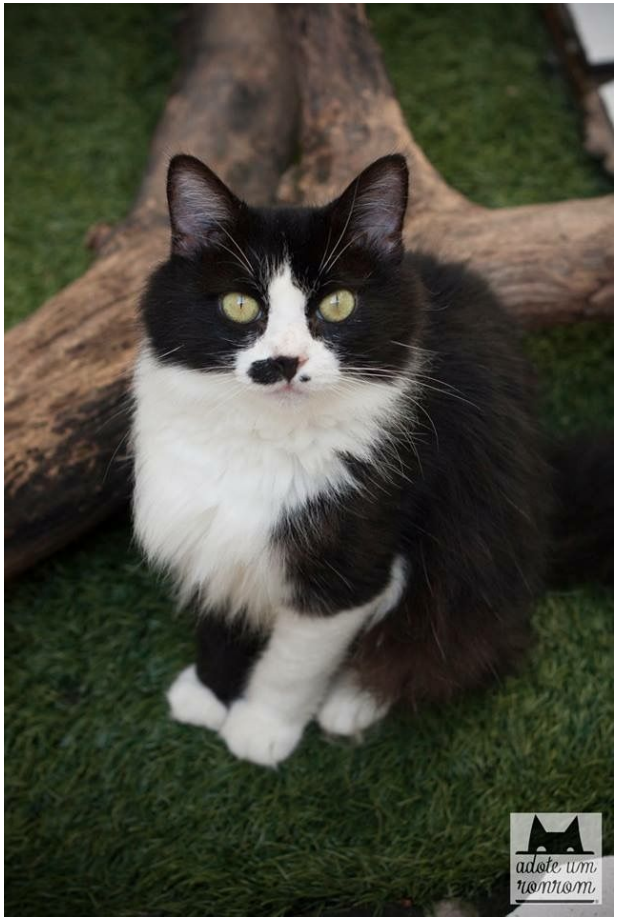
- Tarefa: identificar se o e-mail é spam ou não
- Performance: porcentagem de acertos da classificação desses e-mails
- Experiência: dados de e-mails classificados manualmente

Está aprendendo se a classificação de e-mails está mais eficiente do que o modo anterior (manual)

Tipos de aprendizado de máquina

- Supervisionado
- Por reforço
- Semi-supervisionado
- Não supervisionado

Margot



Aprendizado supervisionado

- Recebe uma amostra
- Recebe uma identificação na coleta de informações
- Estuda essas relações
- Recebe uma amostra desconhecida e a classifica com base no seu conhecimento prévio

Aprendizado por reforço

- Interage em um ambiente dinâmico com uma determinada tarefa
- Suas ações recebem feedbacks
- Esses feedbacks são armazenados e reutilizados

Aprendizado semi-supervisionado

- Recebe uma amostra
- Recebe uma identificação para alguns dados
- Estuda a amostra
- Recebe uma amostra desconhecida e a classifica com base no seu conhecimento prévio

Aprendizado não supervisionado

- Recebe uma amostra
- Analisa a amostra
- Identifica um padrão entre os dados
- Classifica os dados com base no padrão encontrado

O quê tem por trás desse levantamento de padrões?

ESTATÍSTICA

Vamos vivenciar um pouco?

Preparando o Python

Primeiro vamos instalar algumas bibliotecas de pré requisito:

```
[sudo] pip install numpy scipy
```

E agora a própria biblioteca de machine learning:

```
[sudo] pip install scikit-learn
```

O quê vamos classificar?

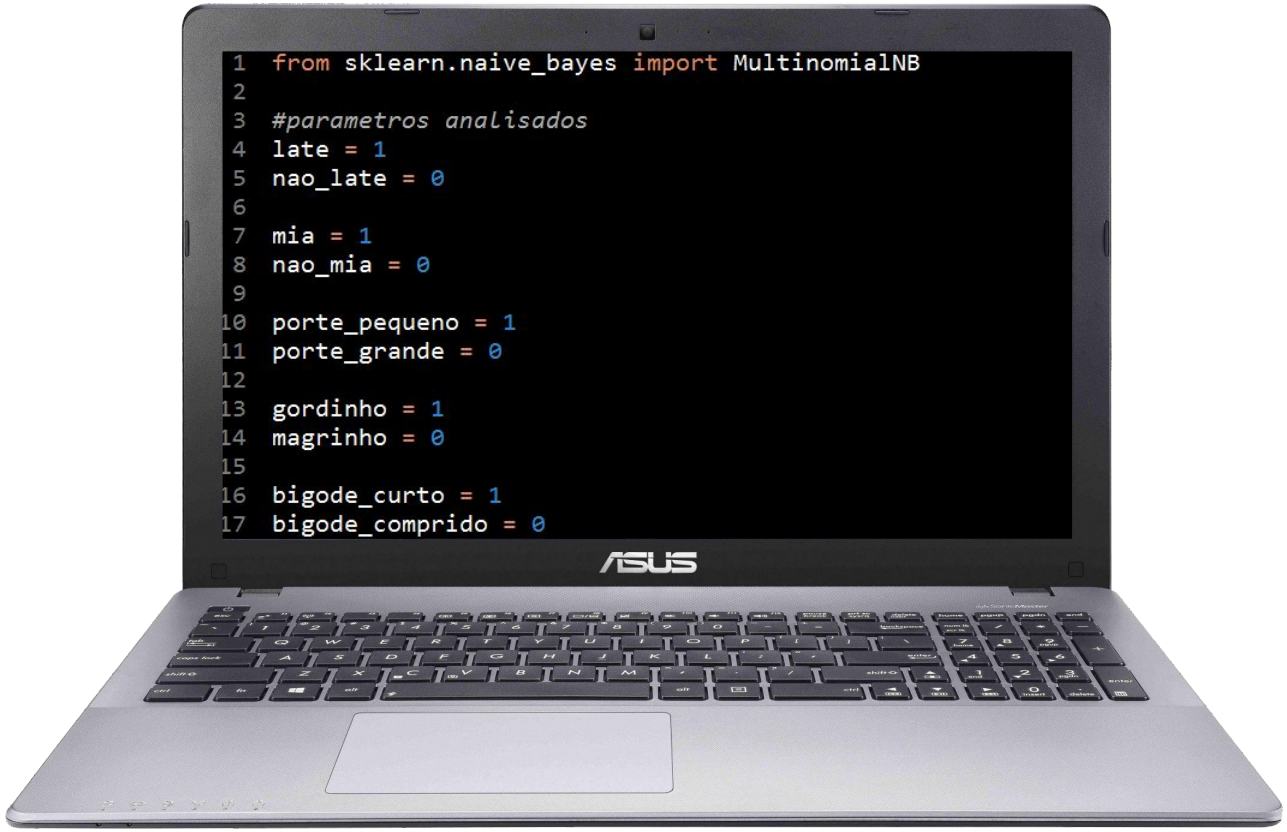
Gatos

- Porte pequeno
- Bigodes longos
- Peso variado
- Mia

Cachorros

- Porte variado
- Bigodes curtos
- Peso variado
- Late

Separando as características em código



```python
from sklearn.naive_bayes import MultinomialNB

#parametros analisados
late = 1
nao_late = 0

mia = 1
nao_mia = 0

porte_pequeno = 1
porte_grande = 0

gordinho = 1
magrinho = 0

bigode_curto = 1
bigode_comprido = 0
```

Atribuindo características a um animal

```
# lista de características de um gatinho
animal_1 = [porte_pequeno, bigode_comprido,
            gordinho, mia, nao_late         ]
# lista de características de um cachorrinho
animal_2 = [porte_pequeno, bigode_curto,
            gordinho, nao_mia, late     ]
```

Vamos usar o modelo supervisionado, nesse caso vamos passar uma identificação para o modelo

Separando em “animais” e “identificação”

```
# tags de classificacoes
gato = 1
cachorro = -1

# lista de animais
conjunto_animais = [animal_1, animal_2]

# identificação respectiva
identificaco = [gato, cachorro]
```

Separando em “animais” e “identificação”

```
# criando um modelo
modelo = MultinomialNB()

# treinando ele com base nas informações
# características vs identificacao
modelo.fit(conjunto_animais, identificacao)
```

Recapitulando...

Fornecemos dados e uma identificação.

Nosso código “treinou” ou “vivenciou” essas informações

Agora vamos passar novos dados para que ele analise e classifique com base no que ele aprendeu

Criando outros animais...

```
#animais que eu nao classifiquei ainda
animal_sem_id_1 = [porte_pequeno, bigode_curto,    gordinho, mia,     nao_late   ]
animal_sem_id_2 = [porte_pequeno, bigode_curto,    magrinho, mia,     late       ]
animal_sem_id_3 = [porte_grande,  bigode_curto,    gordinho, nao_mia, late       ]
animal_sem_id_4 = [porte_pequeno, bigode_curto,    gordinho, nao_mia, nao_late   ]
```

Dando zoom pra facilitar....

```
#animais que eu nao classifiquei ainda
animal_sem_id_1 = [porte_pequeno, bigode_curto,
                   gordinho, mia,      nao_late   ]
animal_sem_id_2 = [porte_pequeno, bigode_curto,
                   magrinho, mia,      late       ]
animal_sem_id_3 = [porte_grande,  bigode_curto,
                   gordinho, nao_mia, late        ]
animal_sem_id_4 = [porte_pequeno, bigode_curto,
                   gordinho, nao_mia, nao_late    ]
```

Análise de resultados

```
# previsao do programa
resultado = modelo.predict(conjunto_sem_id)

# resultado esperado do modelo do programa
resultado_esperado = [gato, gato, cachorro, cachorro]

# calculando a taxa de erros
# gato = 1, se estiver certo 1-1 =0
taxa_erros = resultado - resultado_esperado
```

A taxa de erros é relevante para levantarmos a eficiência do código

Análise de eficiência

```python
# armazena os acertos
acertos = [d for d in taxa_erros if d == 0]

#retorna a qtd de acertos
qtd_acertos = len(acertos)

print(qtd_acertos)
```

Análise de eficiência

```
taxa_acertos = 100 * (qtd_acertos / qtd_sem_id)

print(taxa_acertos)
```

Como podemos melhorar a eficiência?

Acrescentando mais informações

```
# lista de características de gatinhos
animal_1 = [porte_pequeno, bigode_comprido, gordinho, mia, nao_late]
animal_2 = [porte_pequeno, bigode_curto,    gordinho, mia, nao_late]
animal_3 = [porte_pequeno, bigode_comprido, magrinho, mia, nao_late]

# lista de cachorrinhos
animal_4 = [porte_pequeno, bigode_curto, gordinho, nao_mia, late    ]
animal_5 = [porte_grande,  bigode_curto, magrinho, nao_mia, late    ]
animal_6 = [porte_grande,  bigode_curto, gordinho, nao_mia, nao_late]
```

Análise de eficiência

```
# tags de classificacoes
gato = 1
cachorro = -1

# lista de animais
conjunto_animais = [animal_1, animal_2, animal_3, animal_4, animal_5, animal_6]

# identificacao respectivamente
identificacao = [gato, gato, gato, cachorro, cachorro, cachorro]
```

Esse é um modelo pequeno e didático. Modificando-o pode chegar a taxa utópica de 100%

Em estatística a amostra de dados deve ser relevante e a taxa de acertos deve estar acima de 95%

Aprendendo Machine Learning com Python

Workshop de ciência de dados

Giana de Almeida

Vídeos interessantes e material de apoio

Conceitos:
https://www.youtube.com/watch?v=_VUHSPB97jo

http://scikit-learn.org/

http://www.numpy.org/

https://www.scipy.org/

https://pandas.pydata.org/

https://www.statsmodels.org/stable/index.html

https://anaconda.org/anaconda/python

Mia

Os gatos das fotos desse slide estão para adoção s2

Mais informações na página “Adote um ronron”

Ou no e-mail: queroumronrom@gmail.com