

## Aula 34 - Projeto final (parte 2)

Bom, na última aula a gente viu que a gente precisa ter as características dos nossos indivíduos, que são as nossas features. Além disso, existem outros conceitos que são muito importantes que a gente conheça para que a gente possa entender e avaliar o nosso modelo de aprendizado de máquina.

### 1.Features.

- São (características) de entrada que são usadas para fazer previsões no nosso modelo, ou seja, são os nossos dados de entrada que correspondem à característica dos nossos indivíduos.

### 2.Itens

- Os itens são as nossas amostras, ou seja, os nossos dados de treino que a gente vai usar para treinar o modelo.

### 3.Treino do estimador

- O treinamento do estimador é o processo de ajustar um modelo de aprendizado de máquina para que ele possa fazer previsões precisas.

### 4.Testes

- Testes são usados para avaliar a precisão de um modelo de aprendizado de máquina. Os testes usam dados de entrada que o modelo nunca viu antes para avaliar a capacidade do modelo de fazer previsões precisas.

### 5.Taxa de acerto

- A taxa de acerto é uma medida da precisão de um modelo de aprendizado de máquina. Ela representa a proporção de previsões corretas que o modelo fez em relação ao número total de previsões.

### 6.Otimização

- A otimização é o processo de ajustar os parâmetros do modelo de aprendizado de máquina para que ele possa fazer previsões mais precisas.

### 7.Previsão

- A previsão é o processo de usar um modelo de aprendizado de máquina para fazer previsões com base em novos dados de entrada. A previsão é a principal tarefa de um modelo de aprendizado de máquina, e é por isso que é importante avaliar a taxa de acerto do modelo.

Agora que a gente conhece esses conceitos, a gente pode começar a fazer o nosso algoritmo.

Então, a gente quer resolver um clássico problema de classificação, que é de classificar dois tipos de animais, cachorro e gato, e o que o nosso modelo de aprendizado de máquina vai fazer? Ele vai tentar prever que tipo de animal é baseado nas características que a gente definiu no nosso modelo de teste, ou seja, depois que a gente treinar o algoritmo ele vai ser capaz de reconhecer novos dados que são desconhecidos

E, como eu já disse antes, para que a gente possa usar um algoritmo de classificação a gente não precisa criar esse código do zero. Na verdade, a gente pode usar uma biblioteca que já implementa esse código, e além dele, vários outros, como para testar o nosso modelo, ver a precisão e dentre outras coisas.

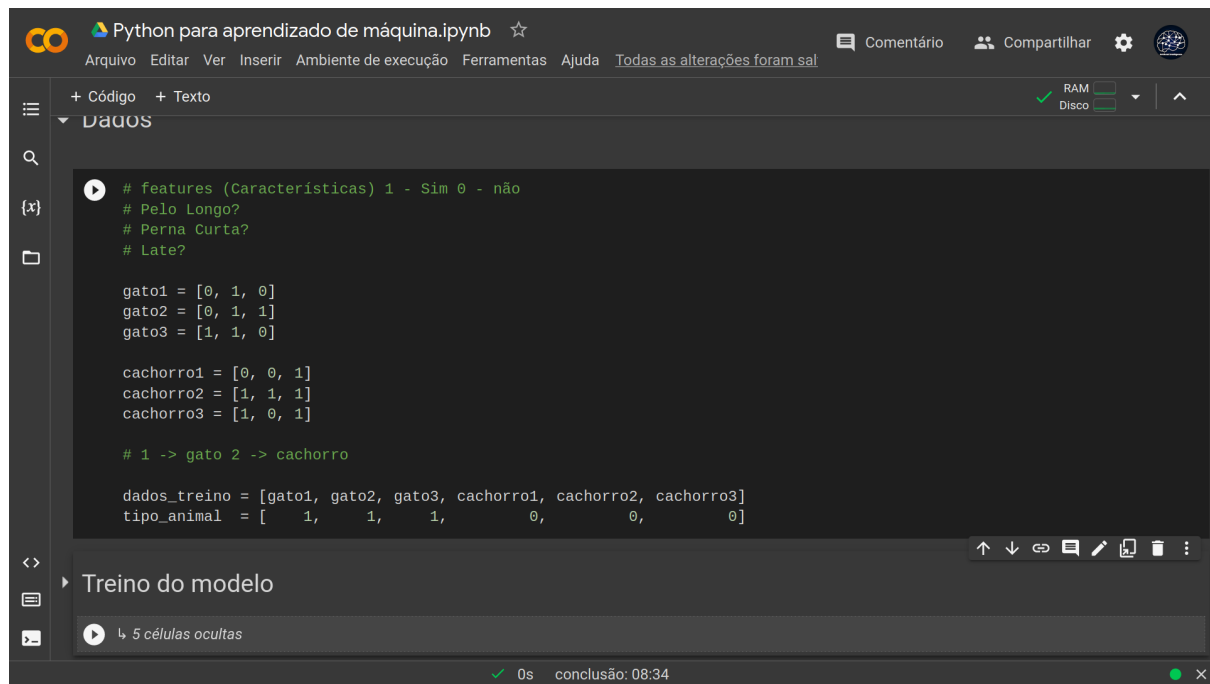
E a biblioteca que a gente vai usar pra isso é a biblioteca chamada Scykit Learn.

E dessa biblioteca a gente vai importar uma classe chamada LinearSVC. Para que essa classe serve?

Basicamente vai ser ela quem vai permitir que a gente treine o nosso modelo baseado no aprendizado supervisionado, ou seja, com base nos dados rotulados que a gente vai passar pro nosso algoritmo.

A gente pode usar o pandas como a nossa base de dados, na verdade ele geralmente é utilizado, mas como a gente vai trabalhar com poucos dados, e é só pra fins didáticos, a gente vai criar uma lista contendo os nossos dados.

A gente pode dizer até, que esses dados são as nossas FEATURES.



```
# features (Características) 1 - Sim 0 - não
# Pelo Longo?
# Perna Curta?
# Late?

gato1 = [0, 1, 0]
gato2 = [0, 1, 1]
gato3 = [1, 1, 0]

cachorro1 = [0, 0, 1]
cachorro2 = [1, 1, 1]
cachorro3 = [1, 0, 1]

# 1 -> gato 2 -> cachorro

dados_treino = [gato1, gato2, gato3, cachorro1, cachorro2, cachorro3]
tipo_animal = [1, 1, 1, 0, 0, 0]
```

```
# 1 -> gato 0 -> cachorro
```

```
# 1 -> gato 0 -> cachorro
```

A gente vai ter o seguinte, o número 1 vai representar o gato e o 0 vai representar o cachorro. Então nós criamos duas tabelas que representam os nossos dados. Agora a gente pode criar uma variável e chamar de `dados_treino`, porque são os dados que a gente vai usar para treinar o nosso algoritmo.

E além disso, cada um desses dados, eu sei a que classe ele pertence, isto é, se ele pertence à classe gato ou à classe cachorro.

Então a gente pode criar uma outra variável para guardar a classe de cada animal.

```
tipo_animal = [1, 1, 1, 0, 0, 0]
```

Então agora nós queremos fazer uma estimativa da classe do tipo de dado. E para isso a gente vai agora importar a nossa biblioteca e a classe para trabalhar com a aprendizagem supervisionada.

E da biblioteca `sklearn` a gente vai importar um estimador chamado `LinearSVC`.

```
from sklearn.svm import LinearSVC
```

Essa classe, `LinearSVC` é basicamente uma implementação do algoritmo de aprendizagem supervisionada.

Então, da mesma forma como a gente aprendia quando éramos criança, o nosso modelo de aprendizagem supervisionada ele vai aprender com dados que a gente vai passar pra ele. E para isso, a gente vai usar uma função chamada *fit* que vai fazer o nosso modelo se adaptar aos nossos dados. Essa função vai preencher o nosso modelo com os nossos dados.

Como a gente tá trabalhando com aprendizagem supervisionada, a gente precisa dizer quais são as classes dos nossos animais, da mesma maneira que a gente aprendeu quando éramos criança. A gente não sabia o que era um gato, e os nossos pais mostravam pra gente o que era um gato, vários gatos, até que um dia a gente foi capaz de reconhecer um gato por nós mesmos. Da mesma maneira vai funcionar o nosso algoritmo.

```
modelo = LinearSVC()

modelo.fit(dados_treino, tipo_animal)
```

Após a gente rodar esse código, basicamente a gente treinou o nosso modelo com base nos nossos dados rotulados.

Agora, a gente quer saber um novo animal, um animal desconhecido, a gente quer que o nosso algoritmo faça uma estimativa do que ele acha que é esse animal.

```
animal_desconhecido = [1, 1, 1]
```

Nesse caso, a gente sabe que esse animal representa um cachorro. Então a gente vai passar esse novo animal desconhecido para o nosso modelo usando a função *predict*, que é a função que a gente quer que o algoritmo vai tentar prever que tipo de animal é esse, a que classe ele pertence.

```
[12] animal_desconhecido = [1, 1, 1]
0s

[13] modelo.predict([animal_desconhecido])
0s

array([0])
```

Notem que a gente passou o nosso animal dentro de uma lista, isso acontece porque a função *predict* espera uma lista como parâmetro.

Isso significa então que a gente pode passar mais de um animal como parâmetro.

```
animal_desconhecido1 = [1, 1, 1]
animal_desconhecido2 = [1, 1, 0]
animal_desconhecido3 = [0, 1, 1]

testes = [animal_desconhecido1, animal_desconhecido2, animal_desconhecido3]

modelo.predict(testes)

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

A gente pode ver que a nossa saída foi de cachorro, gato, gato.

Mas na verdade, eu já sabia quais os tipos de animais eram esses, e na verdade eram cachorro, gato, cachorro. Isso significa então que o nosso modelo errou 1 dos animais.

```
[0, 1, 0]
```

Essa deveria ser a saída do nosso algoritmo.

Se a gente for estimar uma precisão para ele, ou seja, uma taxa de acerto, então o nosso modelo tem uma precisão de 66,66%. E a gente pode fazer uma maneira de receber essa precisão automaticamente.

```
previsoes = modelo.predict([animal_desconhecido1, animal_desconhecido2, animal_desconhecido3])
```

usando uma variável para guardar as previsões do nosso modelo, podemos então comparar com o nosso valor real.

```
previsoes == testes_classe
array([ True,  True, False])
```

Se a gente usar a função `sum()` a gente pode obter o resultado de animais que foram acertados.

```
corretos = (previsoes == testes_classe).sum()
corretos
2
```

Então a gente pode usar a quantidade dos nossos dados para verificar a taxa de acerto:

```
total = len(testes)
taxa_de_acerto = corretos/total
print("Taxa de acerto: ", taxa_de_acerto * 100)
Taxa de acerto:  66.66666666666666
```

Mas, como bom programador, a gente quer fazer a menor quantidade de código possível. Então, a gente pode usar uma implementação que já está na biblioteca `sklearn`.

```
from sklearn.metrics import accuracy_score
taxa_de_acerto = accuracy_score(testes_classe, previsoes)
print("Taxa de acerto: ", taxa_de_acerto * 100)
Taxa de acerto:  66.66666666666666
```

Finalizamos aqui o nosso curso de introdução ao aprendizado de máquina fazendo uma pequena implementação de um algoritmo de classificação de aprendizado supervisionado usando Python e a biblioteca chamada scikit learn.

A área de machine learning e inteligência artificial é imensa, e com muitos conceitos complexos e abstratos, que demandam muito mais do que um curso introdutório para serem compreendidos, quem sabe no futuro, a gente não consiga disponibilizar um curso mais avançado, abordado por exemplo, conceitos mais profundos como Redes neurais ou até mesmo fazendo um projeto mais avançado para o reconhecimento de imagens, ou uma IA para aprender a jogar um jogo, programando cada linha do código e entendendo o seu funcionamento.

A área de inteligência artificial é o futuro da tecnologia, e já podemos dizer que estamos vivendo esse momento de disrupção da tecnologia, quem aí não deve ter ouvido falar do Chat Gpt e o quão revolucionário ele é?

Mas por essa aula é isso, muito obrigado a todos que nos assistiram até aqui e até o nosso próximo curso!