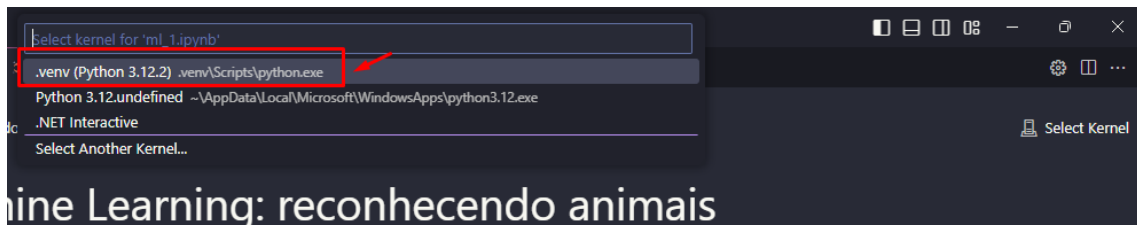


ML_01

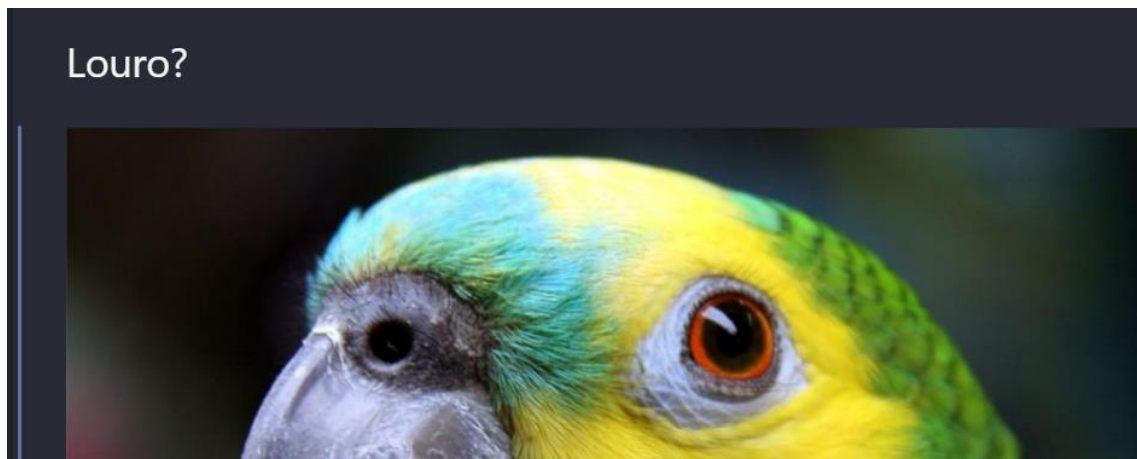
Crie um novo notebook ml_01. Nele insira um título e uma foto de um papagaio.



Em **Select Kernel** escolha a venv ativada.



Coloque um título na imagem







E na sequência coloque de uma galinha

Cocoricó ?









Usando seus conhecimentos de Markdown e inteligência artificial crie uma tabela conforme abaixo:

^ Situação Problema

Imagem	galinha_papagaio	tem bico	risco de extinção	faz cocoricó
	1	Sim	Não	Sim
	0	Sim	Sim	Não
	1	Sim	Não	Sim
	0	Sim	Sim	Não

Essa tabela nos ajudará a entender o problema de aprendizado de máquina e classificar se um determinado animal é uma galinha ou um papagaio.

Crie uma tabela completa:

Imagem	galinha_papagaio	tem bico	risco de extinção	faz cocoricó
	0	Sim	Não	Sim
	1	Sim	Sim	Não
	0	Sim	Não	Sim
	1	Sim	Sim	Não
	0	Sim	Sim	Não
	1	Sim	Sim	Sim

Agora crie uma lista com os papagaios

```
# Papagaio

# features [1 sim, 0 nao]
# tem bico ?
# risco extinção ?
# faz cocoricó ?

papagaio1 = [1, 1, 0]
papagaio2 = [1, 1, 0]
papagaio3 = [1, 1, 1]
```

✓ 0.0s

Crie uma lista com as galinhas:

```
# Galinha

# features [1 sim, 0 nao]
# tem bico ?
# risco extinção ?
# faz cocoricó ?

galinha1 = [1, 0, 1]
galinha2 = [1, 0, 1]
galinha3 = [1, 1, 1]
```

✓ 0.0s

Crie agora os dados:

```
dados = [papagaio1, papagaio2, papagaio3, galinha1, galinha2, galinha3]
```

[4] ✓ 0.0s

Crie também as classes

```
dados = [papagaio1, papagaio2, papagaio3, galinha1, galinha2, galinha3]
classes = [1, 1, 1, 0, 0, 0]
```

[5] ✓ 0.0s

Nosso objetivo é a classificação. Temos dois classes: papagaio e galinha.

Mostre os nossos dados:

```
dados
```

[6] ✓ 0.0s

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

Instale o Scikit-Learn

Treinamento do Modelo

```
! pip install scikit-learn
```

[8] ↻ 8.1s

Coloque para treinar

```
from sklearn.svm import LinearSVC

modelo = LinearSVC()
```

[10] ✓ 0.0s

```
modelo.fit(dados, classes)
```

[11] ✓ 0.0s

... `LinearSVC` ⓘ ⓘ
`LinearSVC()`

Vamos prever um animal

Prever um animal

```
# features [1 sim, 0 nao]
# tem bico ?
# risco extinção ?
# faz cocoricó ?

# tem bico , esta em extinção e não faz cocoricó

animal_misterioso = [1, 1, 0]
previsao = modelo.predict([animal_misterioso])
```

[14] ✓ 0.0s



previsao

[15] ✓ 0.0s

... array([1])

Faça uma previsão de uma galinha agora

```
# features [1 sim, 0 nao]
# tem bico ?
# risco extinção ?
# faz cocoricó ?

# tem bico , esta em extinção e não faz cocoricó

animal_misterioso2 = [1, 0, 1]
previsao2 = modelo.predict([animal_misterioso2])
```

[16] ✓ 0.0s



previsao2

[17] ✓ 0.0s

... array([0])

Estamos imersos no domínio da Classificação, uma técnica essencial no campo do aprendizado de máquina, cujo objetivo primário é a categorização ou rotulagem de dados em diferentes

classes ou categorias pré-definidas. Um exemplo comum seria a distinção entre emails como "spam" ou "não spam", ou a identificação de elementos visuais em imagens, como a presença de um "gato" ou um "cachorro".

Nosso projeto foi inaugurado com a definição minuciosa das características de papagaios e galinhas, empregando o algoritmo LinearSVC para proceder com a classificação desses seres.

Compreendendo o LinearSVC

O LinearSVC se insere no arcabouço das Máquinas de Vetores de Suporte (SVM), sendo um algoritmo que utiliza uma função linear para discriminar os dados em classes distintas. Vamos elucidar seu funcionamento de forma acessível:

- **Coleta de Dados:** O primeiro passo envolve a compilação de dados sobre papagaios e galinhas. Esses dados podem abranger características como peso, altura, tonalidade das penas, comprimento das asas, entre outros. Cada amostra em nosso conjunto de dados é descrita por múltiplas características e acompanhada de um rótulo que especifica se o animal é um papagaio ou uma galinha.
- **Treinamento do Modelo:** O LinearSVC examina essas características e busca delinear a melhor fronteira decisória — uma linha, ou, em cenários com múltiplas características, um hiperplano — que segregue as classes de papagaios e galinhas. A linha ideal é aquela que maximiza a margem entre os exemplos mais próximos de ambas as classes, os chamados vetores de suporte.
- **Classificação de Novas Instâncias:** Após o treinamento, quando o modelo já definiu a linha divisória, ele passa a ser capaz de categorizar novos dados. Se uma nova amostra cair de um lado da linha, será classificada como papagaio; se situar-se do outro lado, será identificada como galinha.

Esse entendimento do funcionamento do LinearSVC e sua capacidade de separar e classificar dados fornecerá uma base sólida para a aplicação desse conhecimento em diversos contextos analíticos. Incentivo-o a continuar explorando e exercitando essas técnicas para aprimorar suas habilidades com este poderoso algoritmo.

Você agora deve calcular a acurácia:

Calculando a Acurácia do Modelo

Dados e Rótulos

```
# dados de treino x
treino_x = [papagaio1, papagaio2, papagaio3, galinha1, galinha2, galinha3]
# rotulos (classes) de treino y
treino_y = [1, 1, 1, 0, 0, 0]
```

[19] ✓ 0.0s

```
# animais misteriosos - para teste
misterio1 = [1, 1, 1]
misterio2 = [1, 1, 0]
misterio3 = [0, 1, 1]
```

[20] ✓ 0.0s

```
# previsões - dados de testes
teste_x = [misterio1, misterio2, misterio3]
# prevendo os testes
previsoes = modelo.predict(teste_x)
```

[21] ✓ 0.0s

```
# rótulos (classes) de teste (avaliação - aqui é a sua provinha)
teste_y = [0, 1, 1]
```

[22] ✓ 0.0s

```
# importando módulo que já traz a função de acurácia encapsulada
from sklearn.metrics import accuracy_score

# definindo uma variável para receber o retorno da função acurácia * 100 para porcentagem
taxa_de_acerto = accuracy_score(teste_y, previsoes) * 100

# mostrando a acurácia
print(f"Acurácia: {taxa_de_acerto:.2f}%")
```

[32] ✓ 0.0s

... Acurácia: 66.67%

... **Acurácia: 66.67%**

Laboratório Concluído – THE END