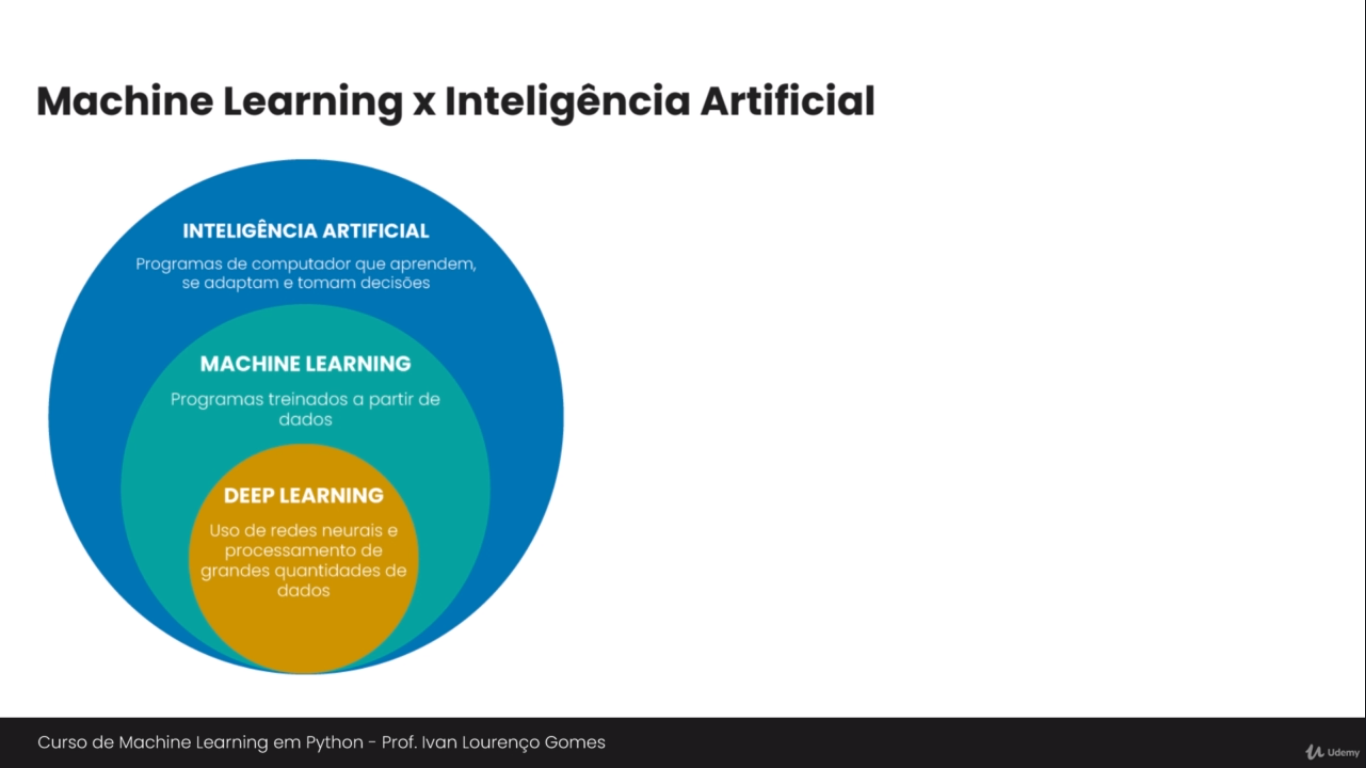
**Curso Machine Learning:**

**1 –** Divisão da inteligência artificial. Ex:



**2 –** Instalações necessárias, pip install scipy, pip install sklearn, pip install ipython, pip install ipython[notebook];

**3 –** Rodando jupyter para python, cmd->jupyter notebook;

**Modelos classificatórios:**

**Iris dataset:**

**1 –** Importando para dentro do projeto o “íris dataset”. Ex:

from sklearn.datasets import load\_iris

iris = load\_iris()

print(iris.data)

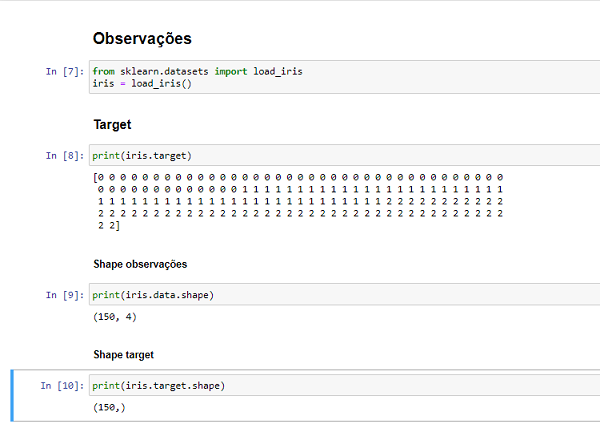
**2-** Para ver os resultados de uma íris dataset usamos “target” (sempre numéricos). Ex:

from sklearn.datasets import load\_iris

iris = load\_iris()

print(iris.data)

print(iriis.target)



**KNN:**

**1 -** Importação do KNN:

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n\_neighbors=1)

**2 -** Treinar a máquina

knn.fit(x,y)

OUT KNeighborsClassifier(n\_neighbors=1)

**3 -** Fazer previsões

species = knn.predict([[5.9,3,5.1,1.8],[5.9,3,5.1,1.8]])[0]

print(iris.target\_names[species])

OUT virginica.

**4 –** Código completo com íris e knn. Ex:

from sklearn.datasets import load\_iris

from sklearn.neighbors import KNeighborsClassifier

from sklearn import metrics

#import matplotlib.pyplot as plt

iris = load\_iris()

x = iris.data

y = iris.target

k = 1

valores = {}

for i in range(k,26):

knn = KNeighborsClassifier(n\_neighbors=k)

knn.fit(x,y)

previsoes = knn.predict(x)

acertos = metrics.accuracy\_score(y,previsoes)

print(k, str(int(acertos\*100))+''.join('%'))

valores[k] = acertos

#print(valores[k])

k = k + 1

**Modelo de regressão logística:**

**1 –** Modelo de regressão logística e uma outra forma de fazer previsões assim como knn. Ex:

from sklearn.linear\_model import LogisticRegression

logreg = LogisticRegression()

logreg.fit(x\_train, y\_train)

previsoes\_logreg = logreg.predict(x\_test)

acertos\_logreg = metrics.accuracy\_score(y\_test, previsoes\_logreg)

print(acertos\_logreg).

**Previsão Quantitativa:**

**1 –** Instalar módulos, pip install pandas (leitura dos dados), pip install seaborn (mais opções de manipulação para textos);

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn import metrics

import numpy as np

publi = pd.read\_csv('http://faculty.marshall.usc.edu/gareth-james/ISL/Advertising.csv', index\_col=0)

publi.head()

tv = 'TV'

radio = 'radio'

news = 'newspaper'

x = publi[[tv,radio,news]]

y = publi['sales']

x\_train, x\_test, y\_train,y\_test = train\_test\_split(x,y,test\_size=0.3, random\_state=5)

reglin = LinearRegression()

reglin.fit(x\_train,y\_train)

y\_prev = reglin.predict(x\_test)

print(np.sqrt(metrics.mean\_squared\_error(y\_test, y\_prev)))

x = publi[[radio,news]]

y = publi['sales']

x\_train, x\_test, y\_train,y\_test = train\_test\_split(x,y,test\_size=0.3, random\_state=5)

reglin = LinearRegression()

reglin.fit(x\_train,y\_train)

y\_prev = reglin.predict(x\_test)

print(np.sqrt(metrics.mean\_squared\_error(y\_test, y\_prev)))

x = publi[[tv,news]]

y = publi['sales']

x\_train, x\_test, y\_train,y\_test = train\_test\_split(x,y,test\_size=0.3, random\_state=5)

reglin = LinearRegression()

reglin.fit(x\_train,y\_train)

y\_prev = reglin.predict(x\_test)

print(np.sqrt(metrics.mean\_squared\_error(y\_test, y\_prev)))

x = publi[[tv,radio]]

y = publi['sales']

x\_train, x\_test, y\_train,y\_test = train\_test\_split(x,y,test\_size=0.3, random\_state=5)

reglin = LinearRegression()

reglin.fit(x\_train,y\_train)

y\_prev = reglin.predict(x\_test)

print(np.sqrt(metrics.mean\_squared\_error(y\_test, y\_prev)))

**Reconhecimento de imagens:**

**1 –** SVM, ex:

from sklearn import datasets

digits = datasets.load\_digits()

print(digits.data.shape)

print(digits.target.shape)

print(digits.images[0])

from sklearn.model\_selection import train\_test\_split

x = digits.data

y = digits.target

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y,test\_size=0.20, random\_state=5)

from sklearn import svm

from sklearn import metrics

classifier = svm.SVC()

classifier.fit(x, y)