Universidade Federal do Rio de Janeiro
Bacharelado em Ciência da Computação
Inteligência Artificial

Relatório:
Aprendizagem de Máquina com “Iris” dataset

Tainá da Silva Lima - DRE: 116165607
Rio de Janeiro
2019

SUMÁRIO

I. Introdução……..……...…….…..…..2

II. Objetivo…….....……....….3

III. Metodologia……..……....…….…....4

IV. Resultados…….....……..…...6

V. Conclusões……..……...…….....8

VI. Referências bibliográficas……....……....……9

I. INTRODUÇÃO

Este relatório é referente ao trabalho da disciplina de Inteligência Artificial, ministrada pelo Prof. Mário Benevides. O trabalho foi passado com o intuito de colocar-se em prática os ensinamentos dados em aula sobre aprendizagem de máquina. Sendo assim, este consiste em utilizar o dataset “Iris” fornecido para tal objetivo. Para resolução deste trabalho foi aplicado o algoritmo de machine learning perceptron (PLA) e seu código fonte foi feito usando a linguagem Python.

O propósito deste documento é relatar o objetivo do trabalho, a metodologia utilizada, bem como quais foram os resultados encontrados e as conclusões geradas por estes.

II. OBJETIVO

O objetivo deste trabalho é, utilizando algum algoritmo de aprendizagem de máquina apresentado em aula e baseado num conjunto de dados de entrada, prever qual é a classe da planta iris: setosa, versicolor ou virginica.

O dataset inicial possui 150 instâncias, as quais são divididas em código nas que serão utilizadas para treinar e as que servirão como teste no final. Este dataset possui informações sobre 4 atributos: altura e largura da sépala, altura e largura da pétala e a classe da planta. Foi utilizado somente os 4 primeiros atributos para prever o último.

III. METODOLOGIA

Para solucionar este problema, foi utilizado o algoritmo perceptron (PLA). Além disso, o algoritmo foi criado usando a linguagem de programação Python (Versão 3.x) e algumas de suas bibliotecas tais como Numpy, OS e matplotlib.

A. PLA

Basicamente o PLA (Perceptron Learning Algorithm) é um algoritmo de aprendizado de máquina supervisionada em que dado um conjunto de dados, ele encontra um hiperplano (uma reta no caso de 2 dimensões) que divide (classifica) tais dados em dois grupos.

O algoritmo começa com um vetor normal ao hiperplano w, que pode ser aleatório. Conforme os dados vão sendo lidos (cada dado é um vetor), os valores dos termos de w são modificados de maneira que para este valor lido, a classificação dele através de w esteja correta. Como tal ação pode gerar erros em "dados passados", todo o dataset deve ser revisto para que o w esteja gerando as saídas corretas.

Esse correção do vetor dos pesos, w, é feita até que, no meu algoritmo, todos as saídas geradas por ele ao aplicá-lo nas entradas estejam corretas, ou até que um limite pré-fixado de iterações seja atingido. Essa última condição foi usada para que, caso não tenha um hiperplano que divida as instâncias perfeitamente, o algoritmo não entre em loop infinito.

B. O algoritmo

Este programa foi criado baseado nos conceitos apresentados em sala de aula e escrito nos slides sobre aprendizado de máquina disponibilizado aos alunos. Sendo assim, foi criado uma função "perceptron(xTrDataS,yTrDataS)" que retorna o w (os pesos) que geram o hiperplano que divide os dados. Além disso, há a função "classification(wFinalS,wFinalVe,xTeDataS,yTeDataS,testData,choice)" que serve para separar os dados de fato, retornando 3 matrizes com as instâncias que supostamente são de cada tipo.

Ademais, foi criado funções para o tratamento e leitura dos dados para o formato de matriz e outra função “finalTest” onde é verificado, para as entradas de teste, qual seria a classe dessa planta de entrada previsto pela perceptron. Nessa mesma função também é calculado o número de acertos e erros para o grupo de teste.

IV. RESULTADOS

Como são 3 tipos de planta e o perceptron só separa em dois grupos, foi necessário executar o algoritmo duas vezes: uma para separar um tipo do resto e, desse resto, dividir nos dois tipos restantes.

O algoritmo começa com a divisão setosa/não-setosa e faz isso com excelência com uma taxa de acertos de 100%, ou seja, ele acerta todos casos que são de fato a iris setosa. O problema começou quando fomos separar as não-setosas em versicolor e em virginica.

Utilizando a versicolor como parâmetro de dicotomização, a taxa de acerto já diminui consideravelmente, acertando muito pouco em alguns casos de entrada de treino, algo por volta de 50% ou acertando uma quantidade razoável, aproximadamente 80%.

```
Resultados:
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Errou! Não é virginica!
Acertou! É versicolor!
Errou! Não é virginica!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É virginica!
Acertou! É virginica!
Errou! Não é versicolor!
Acertou! É virginica!
Acertou! É virginica!
Errou! Não é versicolor!
Acertou! É virginica!
Errou! Não é versicolor!
Acertou! É virginica!
Acertou! É virginica!
Acertou! É virginica!
Acertou! É virginica!
Acertou! É virginica!
O número de acertos foi: 25 e os erros: 5
A taxa de acertos foi: 83.33333333333334%
```

Figura 2: Saída do perceptron, versicolor como parâmetro

Foi percebido então, que utilizando a virginica como parâmetro para a segunda separação gerava melhores resultados do que quando era a versicolor, chegando a até, em alguns casos, a 100% de acertos quando o teste é feito.

```
Resultados:
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É setosa!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É versicolor!
Acertou! É virginica!
Acertou! É virginica!
Acertou! É virginica!
Acertou! É virginica!
Acertou! É virginica!
Acertou! É virginica!
Acertou! É virginica!
Acertou! É virginica!
Acertou! É virginica!
Acertou! É virginica!
O número de acertos foi: 30 e os erros: 0
A taxa de acertos foi: 100.0%
```

Figura 3: Saída do perceptron, virginica como parâmetro

O porquê isso ocorre é algo interessante a ser pensado, uma vez que, analisando a disposição dos dados no ponto de vista de apenas dois parâmetros, não se vê muita diferença no nível de dificuldade em traçar uma reta dividindo a partir de versicolor ou a partir de virginica. É provável que no $\mathbb{R}^4$ faça diferença utilizar o primeiro ou o segundo.

V. CONCLUSÕES

Este trabalho teve como objetivo colocarmos em prática os algoritmos de machine learning aprendidos em sala, bem como testar sua eficiência e eficácia. Ele foi ótimo para analisarmos como o PLA funciona. Até então não tinha conhecimentos práticos sobre o assunto, achei interessante programar sobre.

Inicialmente achei que estava tendo resultados muito ruins, uma vez que ao usar o versicolor como parâmetro para realizar a segunda divisão me gerava erros muito grandes em alguns casos. Contudo, após ter pensado em trocar para dicotomizar através do tipo virginica, obtive resultados bem melhores em que a maioria das vezes em que o algoritmo foi executado a taxa de sucesso ficou acima de 80%, chegando 100% como foi exemplificado ao longo deste relatório.

VI. REFERÊNCIAS BIBLIOGRÁFICAS

- https://www.tutorialspoint.com/python/index.htm
- https://docs.python.org/3/tutorial/
- https://en.wikipedia.org/wiki/Perceptron
- https://archive.ics.uci.edu/ml/datasets/iris
- http://web.mit.edu/course/other/i2course/www/vision_and_learning/perceptron_notes.pdf