Aprendizado de Máquina - Relatório

Prática 05 - Árvore de Decisão

1 - Introdução

Árvores de decisão (Decision Trees) são ferramentas que podem ser utilizadas para dar ao agente a capacidade de aprender, bem como para tomar decisões. A idéia de aprendizado é que os perceptos (elementos do agente que percebem o mundo) não seja usado apenas para agir, mas também para aumentar a capacidade do agente de agir no futuro. O aprendizado ocorre na medida que o agente observa suas interações com o mundo e seu processo interno de tomada de decisões. Aprendizado de árvores de decisão é um exemplo de aprendizado indutivo: Cria uma hipótese baseada em instâncias particulares que gera conclusões gerais.

Árvores de decisão são similares a regras if-then. É uma estrutura muito usada na implementação de sistemas especialistas e em problemas de classificação. As árvores de decisão tomam como entrada uma situação descrita por um conjunto de atributos e retorna uma decisão, que é o valor produzido para o valor de entrada. Os atributos de entrada podem ser discretos ou contínuos.

A árvore de decisão chega a sua decisão pela execução de uma seqüência de testes. Cada nó interno da árvore corresponde a um teste do valor de uma das propriedades, e os ramos deste nó são identificados com os possíveis valores do teste. Cada nó folha da árvore especifica o valor de retorno se a folha for atingida.

A maioria das árvores de decisão são geradas de acordo com a entropia, a qual fornece a quantidade de informação que um atributo tem a oferecer sobre a conclusão. Assim, de acordo com a entropia é possível realizar o processo de separação do conjunto de atributos e deste modo gerar a árvore de acordo com aqueles atributos que melhor representam a classe. Contudo, quanto menor for a entropia, mais informação esse atributo tem a oferecer.

2 - Implementação

Inicialmente foi feita uma análise dos dados e realizado um processo de codificação dos atributos, de modos que foi fornecido números entre 0,1 e -1 para poder representá-los. Deste modo, todos os atributos do arquivo “frutas” foi modificado de acordo uma ordem predefinida.

A árvore de decisão foi criada tendo como base o atributo “RISCO”. O processo de separação e construção da árvore se deu pelo cálculo da entropia, sendo que sempre era escolhido aquele atributo que apresentou uma baixa entropia (Maior informação). O cálculo da entropia foi feito seguindo a especificação do slide passado em sala. Assim, inicialmente era escolhido um atributo e então recursivamente era gerado o ramo do atributo escolhido. O algoritmo alternava de atributo quando o mesmo encontra-se uma folha.

Contudo, a maior dificuldade foi compreender o processo de criação da árvore, após entender como acontece o “Split” foi possível concluir a prática.

3 - Resultados

Como resultados obtivemos aqueles atributos que melhor representam o conjunto de dados. Para questões de visualização foi criada uma função que efetua a impressão da árvore, possibilitando assim analisar melhor sua estrutura.

Árvore de decisão gerada.

Abaixo é mostrado a estrutura da árvore e assim os atributos mais representativos do conjunto de dados “frutas”.

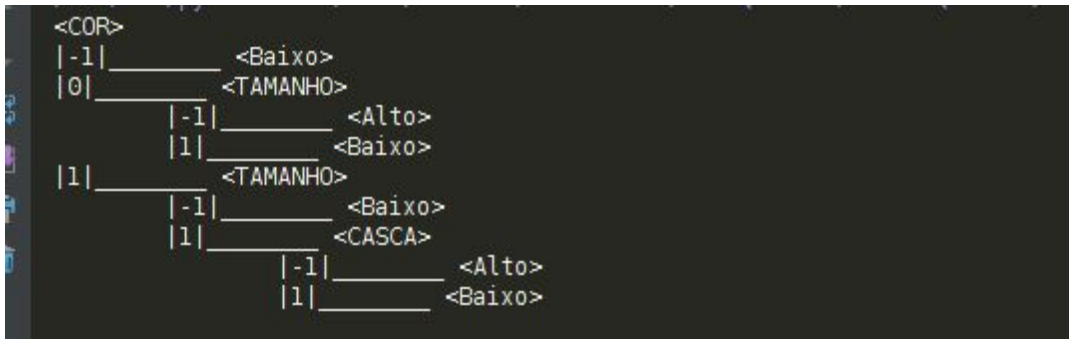
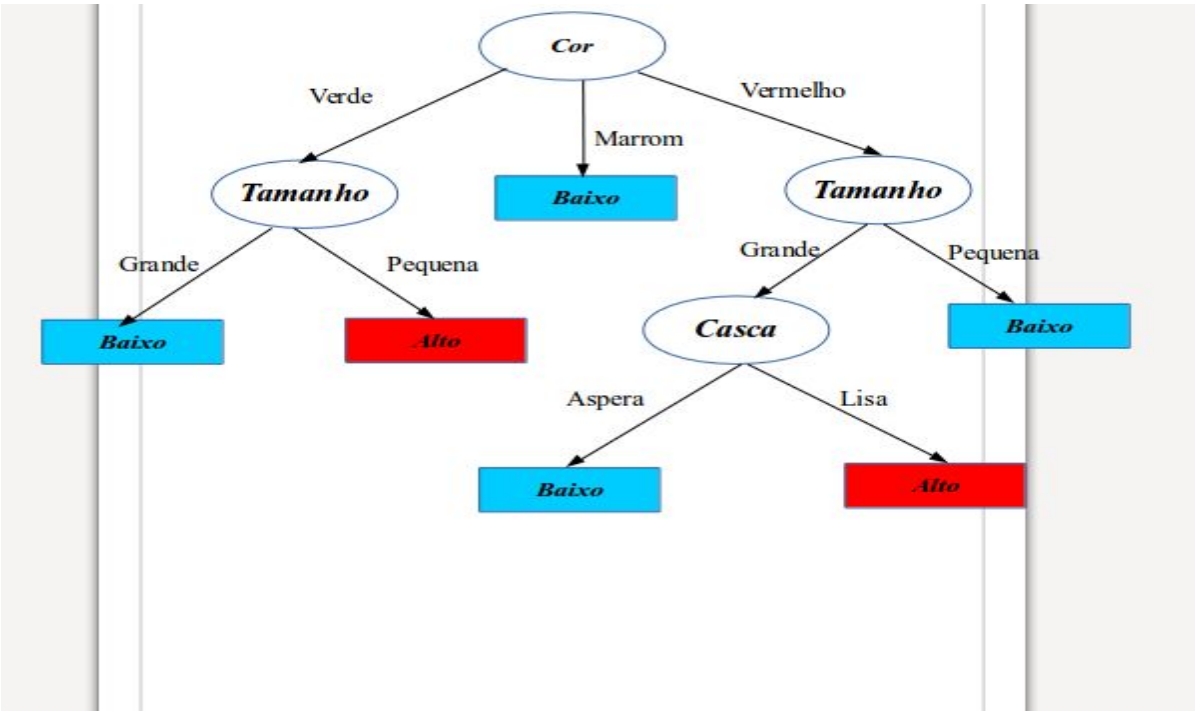


Figura 01: Árvore de Decisão para o conjunto de dados frutas.

Para um entendimento melhor do que a imagem acima significa, na imagem abaixo é apresentado o conteúdo da imagem anterior de um modo mais legível e compreensível.



Portanto, ao realizar uma análise da imagem acima, o resultado obtido com a implementação da árvore de decisão foi satisfatório. Visto que a árvore obtida está muito parecida com a que foi disponibilizada no material de aula, variando apenas no penúltimo nível.

4 - Código Fonte

Estrutura da Árvore

```
# -*- coding:utf-8 -*-

class ArvoreDecisao:

  def __init__(self, nome_atributo = '', valor_atributo = '', filhos = []):
     self.pai = None
     self.nome_atributo = nome_atributo
     self.valor_atributo = valor_atributo
     self.filhos = filhos

     for filho in self.filhos:
        filho.pai = self

  def __str__(self):
     return self.nome_atributo


# Mostra a árvore
def printArvore(arvore, nivel=0):
  print  '<' + str(arvore.nome_atributo) + '>'
  for filho in arvore.filhos:
     print '       ' * nivel + '|' + str(filho.valor_atributo) + '|' + '_________',
     printArvore(filho, nivel+1)
```

Principal

```
# -*- coding:utf-8 -*-
import arvore
import numpy as np
import math

# Retorna um novo conjunto de dados com todos os elementos daquele tipo
def getAtributos(data, coluna, valor):
  atributos = []
  for linha in data:
     if linha[coluna] == valor:
        atributos.append(linha)

  return np.array(atributos)
```

```
#Calcula a entropia de um atributo
def calcularEntropia(data, coluna_atributo, valores_atributo, coluna_classe, valores_classe ):
  valor_atributo = []
  ocorrencias = []
  tamanho = float(len(data))

  for valor in valores_classe:
    ocorrencias.append(getAtributos(data, coluna_classe, valor))

  for valor in valores_atributo:
    probrabilidades = []
    atributo = getAtributos(data,coluna_atributo, valor)
    probOcorrencia = len(atributo)/tamanho
    probrabilidades.append(probOcorrencia)
    for elemento in ocorrencias:
      tamanho_atributo = float(len(atributo))
      if tamanho_atributo != 0:
        qtd_ocorre = len(getAtributos(elemento, coluna_atributo, valor))
        probrabilidades.append(qtd_ocorre/tamanho_atributo)
      else:
        pass
    valor_atributo.append(tuple(probrabilidades))

  entropia = 0
  for elemento in valor_atributo:
    somaElemento = 0
    for i in range(1,len(elemento)):
      if elemento[i] != 0:
        somaElemento += elemento[i]*math.log(elemento[i], 2)
    entropia += elemento[0]*somaElemento
  return -entropia

# Faz uma verificação para saber se é folha
def isFolha(data, coluna):
  a_inicial = data[0,coluna]
  folha = True
  for linha in data:
    if linha[coluna] != a_inicial:
      return not folha
  return a_inicial

# Construindo a árvore
def construirArvore(data, colunaClasse, valoresClasse, valoresAtributos, atributos, valorAtributo=None):
  entropias = []
  folha = isFolha(data, colunaClasse)
  if folha:
    return arvore.ArvoreDecisao(nome_atributo=valoresClasse[folha], valor_atributo=valorAtributo)

  for i in atributos:
    entropias.append(calcularEntropia(data, i, valoresAtributos[i], colunaClasse, valoresClasse))
```

```
  coluna_atributo = entropias.index(min(entropias))
  filhos = []
  for valor in valoresAtributos[coluna_atributo]:
     dados = getAtributos(data, coluna_atributo, valor)
     if dados.size != 0:
        f = construirArvore(dados, colunaClasse, valoresClasse, valoresAtributos, atributos, valor)
        filhos.append(f)

      return  arvore.ArvoreDecisao(nome_atributo=atributos[coluna_atributo],  valor_atributo=valorAtributo,
filhos=filhos)

# Função principal
def main():
  data = np.genfromtxt("frutas", delimiter=",")
  # Referente a classe RISCO
  colunaClasse = 4
  #Valores possíveis da classe para classificação
  valoresClasse = {-1: "Baixo", 1: "Alto"}
  #Para cada atributo define sua quantidade e seus valores possiveis
  valoresAtributos = [[-1,1], [-1,0,1], [-1,1], [-1,1]]
  atributos = {0 : 'CASCA', 1 : 'COR', 2 : 'TAMANHO', 3 : 'POLPA'}

  arvore_decisao = construirArvore(data, colunaClasse, valoresClasse, valoresAtributos, atributos)
  arvore.printArvore(arvore_decisao)

if __name__ == "__main__":
  main()
```