bayes classification

Lov Dubey

November 4, 2022

1 bayes classification

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage{listings}
\title{Assignment Baysian Network}
\author{shagun.2124mca1045 }
\date{November 2022}
\begin{document}
\large
\maketitle
\section{Introduction}
\begin{lstlisting}
def first_number_index(L):
"""
function to find the index of first number in one line of document
"""
for i,s in enumerate(L):
try:
float(s)
return i
except:
pass
def file_to_cpt(filename):
"""
function to convert text file into conditional porbability table and compute compactness
"""
with open(filename,'r') as f:
# conditional probability table
CPT = {}
# nodes in BN
1
nodes = []
# num of CPT lines
```

```
n_rows = 0
for line in f:
line = line.replace('\r\n','')
if line != 'END':
content = line.split()
print 'Reading node '+ content[0]
nodes.append(content[0])
# index of first number
index = first_number_index(content)
parents = content[1:index]
if 'NONE' in parents:
n_rows += 1
else:
n_rows += 2**len(parents)
CPT[content[0]] = {'parents':parents,'cpt':[float(num) for num in content[index:]]}
#cpt = [content[0],parents,[float(num) for num in content[index:]]]
#CPT.append(cpt)
else:
print filename+' read.'
compactness = 1.*n_rows/(2**len(nodes))
return CPT,n_rows,2**len(nodes),compactness,nodes
def bool_to_var(Bool,Var):
"""
function to transform boolean list into list with variable name
e.g., Bool = [1,0], Var = ['Burglary','Earthquake'], then the output would be ['nBurglary',
"""
new_var = []
for i,j in enumerate(Bool):
if j == 1:
new_var.append('n'+Var[i])
else:
new_var.append(Var[i])
return new_var
def parents_to_index(parents,jd):
"""
function to use parents to get index in cpt
"""
binary = ''
for parent in parents:
if parent in jd:
binary += '0'
2
else:
binary += '1'
return int(binary,2)
def joint_prob(CPT,nodes,out_filename):
```

```
"""
function to compute full joint probability and write result to a txt file
"""
# num of multiplications
n_mul = 0
# num of additions
n_add = 0
# list to store joint distributions
result = []
for i in range(1<<len(nodes)):
# joint probability
joint_prob = 1.
# True/False boolean value
tf = bin(i)[2:]
tf = '0'*(len(nodes)-len(tf))+tf
tf = map(int,list(tf))
jd = bool_to_var(tf,nodes)
for node in jd:
if node[0] != 'n':
parents = CPT[node]['parents']
if 'NONE' in parents:
joint_prob *= CPT[node]['cpt'][0]
n_mul += 1
else:
index = parents_to_index(parents,jd)
joint_prob *= CPT[node]['cpt'][index]
n_mul += 1
else:
parents = CPT[node[1:]]['parents']
if 'NONE' in parents:
joint_prob *= (1.-CPT[node[1:]]['cpt'][0])
n_mul += 1
n_add += 1
else:
index = parents_to_index(parents,jd)
joint_prob *= (1.-CPT[node[1:]]['cpt'][index])
n_mul += 1
n_add += 1
result.append([jd,joint_prob])
with open(out_filename,'w') as f:
3
for line in result:
f.write(str(line[0])+'\t')
f.write(str(line[1])+'\n')
return n_mul,n_add
def p_summary(filename,CPT,n_cpt,n_jd,compactness,n_mul,n_add):
```

```
"""
function to print summary statistics for BN
"""
print CPT
print '-----------------------------------------------------'
print 'Bayesian Network :\t'+filename
print 'Num of lines in joint distribution:\t'+str(n_jd)
print 'Num of CPT lines :\t'+str(n_cpt)
print 'Compactness :\t'+str(compactness)
print 'Num multiply operations :\t'+str(n_mul)
print 'Num add operations :\t'+str(n_add)
print '-----------------------------------------------------'
print '\n'
def comp_joint_dist(filename):
# construct CPT from source file
CPT,n_cpt,n_jd,compactness,nodes = file_to_cpt(filename)
# file to store full joint distribution
out_filename = 'out_'+filename
# compute full joint distribution
n_mul,n_add = joint_prob(CPT,nodes,out_filename)
# print summary information
p_summary(filename,CPT,n_cpt,n_jd,compactness,n_mul,n_add)
filename = ['examplebayes.txt','bayesnets1.txt','bayesnets2.txt','bayesnets3.txt','bayesnets
for name in filename:
comp_joint_dist(name)
\end{lstlisting}
\end{document}
```