VASAVI COLLEGE OF ENGINEERING (Autonomous)

IBRAHIMBAGH, HYDERABAD – 500 031

Department of Computer Science & Engineering

INNOVATION IN TEACHING

Course Name: Artificial Intelligence **Faculty Name:** Dr. T.Adi Lakshmi

Topic: Bayes Network **Year & Semester: V**I Sem

Innovative Teaching aid/Tool Used: BayesPy

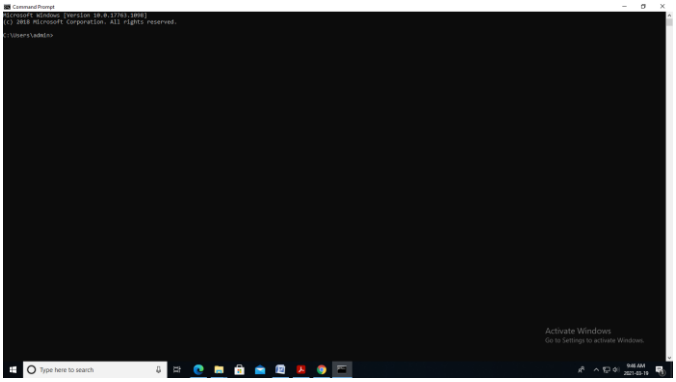
Description of the tool:

BayesPy provides tools for Bayesian inference with Python. The user constructs a model as a Bayesian network, observes data and runs posterior inference. The goal is to provide a tool which is efficient, flexible and extendable enough for expert use but also accessible for more casual users. By removing the tedious task of implementing the variational Bayesian update equations, the user can construct models faster and in a less error-prone way. Simple syntax, flexible model construction and efficient inference make BayesPy suitable for both average and expert Bayesian users. It also supports some advanced methods such as stochastic and collapsed variational inference

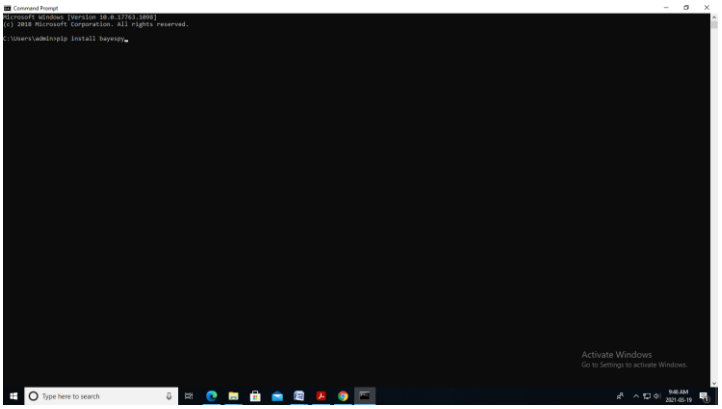
Tool usage in Teaching*:* A Bayes net is a model. It reflects the states of some part of a world that is being modeled and it describes how those states are related by probabilities.

Installing BayesPy

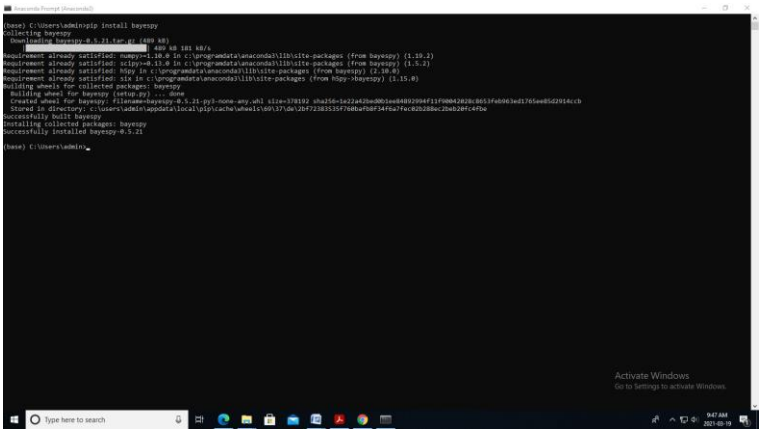
Step 1: Start command prompt (windows) or terminal (ubuntu)



Step 2: type “pip install bayespy”

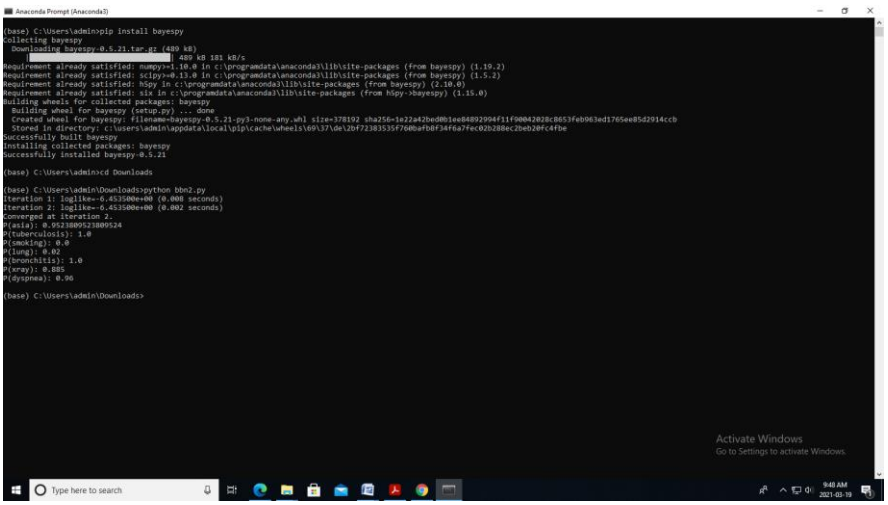


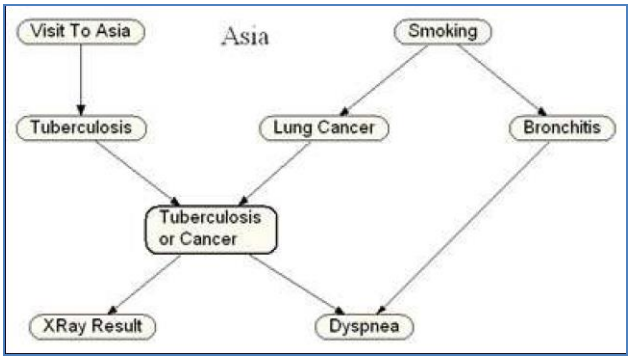
Step3: Installation completes



Step 4: Execute Bayes Belief Network Python code

Python bbn2.py





The above diagrams shows the Bayesian belief network demonstrating Asia problem

CODE Explanation

import numpy as np #For numerical operations

from bayespy.nodes import Categorical, Mixture

#**Categorical Node**: In probability theory and statistics, a categorical distribution (also called a generalized Bernoulli distribution, multinoulli distribution) is a discrete probability distribution that describes the possible results of a random variable that can take on one of *K* possible categories, with the

probability of each category separately specified. The *K*-dimensional categorical distribution is the most general distribution over a *K*-way event; any other discrete distribution over a size-*K* sample space is a special case. The parameters specifying the probabilities of each possible outcome are constrained only by the fact that each must be in the range 0 to 1, and all must sum to 1.

 # Mixture Node for exponential family mixture variables. Node for exponential family mixture variables. The node represents a random variable which is sampled from a mixture distribution. It is possible to mix any exponential family distribution. The probability density function is

$$p(x|z = k, \boldsymbol{\theta}_0, \ldots, \boldsymbol{\theta}_{K-1}) = \phi(x|\boldsymbol{\theta}_k),$$

where ϕ is the probability density function of the mixed exponential family distribution and $\boldsymbol{\theta}_0, \ldots, \boldsymbol{\theta}_{K-1}$ are the parameters of each cluster. For instance, ϕ could be the Gaussian probability density function $\mathcal{N}$ and $\boldsymbol{\theta}_k = \{\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k\}$ where $\boldsymbol{\mu}_k$ and $\boldsymbol{\Lambda}_k$ are the mean vector and precision matrix for cluster k.

from bayespy.inference import VB

#used for inferencing the observations

 # NOTE: Python's built-in booleans don't work nicely for indexing, thus define

 # own variables:

FALSE = 0

TRUE = 1

```
def _or(p_false, p_true):
    """
        Build probability table for OR-operation of two parents
        p_false: Probability table to use if both are FALSE
        p_true: Probability table to use if one or both is TRUE
    """
    return np.take([p_false, p_true], [[FALSE, TRUE], [TRUE, TRUE]], axis=0)
```

#Create node variables as per the diagram and data to be stored

```
asia = Categorical([0.5, 0.5])
tuberculosis = Mixture(asia, Categorical, [[0.99, 0.01], [0.8, 0.2]])
smoking = Categorical([0.5, 0.5])
lung = Mixture(smoking, Categorical, [[0.98, 0.02], [0.25, 0.75]])
bronchitis = Mixture(smoking, Categorical, [[0.97, 0.03], [0.08, 0.92]])
xray = Mixture(tuberculosis, Mixture, lung, Categorical,
               _or([0.96, 0.04], [0.115, 0.885]))

dyspnea = Mixture(bronchitis, Mixture, tuberculosis, Mixture, lung, Categorical,
                  [_or([0.6, 0.4], [0.18, 0.82]),
                   _or([0.11, 0.89], [0.04, 0.96])])
```

 # Mark observations

tuberculosis.observe(TRUE)

smoking.observe(FALSE)

bronchitis.observe(TRUE) # not a "chance" observation as in the original example

 # Run inference

Q = VB(dyspnea, xray, bronchitis, lung, smoking, tuberculosis, asia)

Q.update(repeat=100)

 # Show results

print("P(asia):", asia.get_moments()[0][TRUE])

print("P(tuberculosis):", tuberculosis.get_moments()[0][TRUE])

print("P(smoking):", smoking.get_moments()[0][TRUE])

print("P(lung):", lung.get_moments()[0][TRUE])

print("P(bronchitis):", bronchitis.get_moments()[0][TRUE])

print("P(xray):", xray.get_moments()[0][TRUE])

print("P(dyspnea):", dyspnea.get_moments()[0][TRUE])