CSE 674: Advance Machine Learning (Spring 2019)
Project 1

Objective

This project is to develop probabilistic graphical models (PGMs) to determine probabilities of observations which are described by several variables. Worked with handwriting patterns which are described by document examiners. They can be used to determine whether a particular handwriting sample is common (high probability) or rare (low probability) and which in turn can be useful to determine whether a sample was written by a certain individual. We have consider only the letter pair "th" in this study.

Introduction

Probabilistic Graphical Model (PGM) is a technique of representing Joint Distributions over random variables in a compact way by exploiting the dependencies between them. PGMs use a network structure to encode the relationships between the random variables and some parameters to represent the joint distribution.

There are two major types of Graphical Models: Bayesian Networks and Markov Networks

Bayesian Network: A Bayesian Network consists of a directed graph and a conditional probability distribution associated with each of the random variables. A Bayesian network is used mostly when there is a causal relationship between the random variables. An example of a Bayesian Network representing a student [student] taking some course is shown in Fig 1.

A Bayesian Network consists of a directed graph where nodes represents random variables and edges represent the relation between them. It is parameterized using Conditional Probability Distributions(CPD). Each random variable in a Bayesian Network has a CPD associated with it. If a random varible has parents in the network then the CPD represents P(var|Parvar) i.e. the probability of that variable given its parents.

A BN model B for a set of n variables X = {X1,X2, ... ,Xn} each having a finite set of mutually exclusive states consists of two main components, B ={G; €}. The first component G is a structure that is a directed acyclic graph (DAG) because it contains no directed cycles. The nodes of G correspond to the variables of X, and thus a variable and its corresponding node are usually referred interchangeably. An edge connecting two nodes in G manifests the existence of direct causal influence between the corresponding variables, and the lack of a possible edge in G represents conditional independence (d-separation) between the corresponding variables.
The second component of BN is a set of parameters €, h, that specify all the conditional probability distributions (or densities) that quantify graph edges.
The joint probability distribution for X given a structure G that is assumed to encode this distribution is given using the set of parameters by

$$P(\boldsymbol{X}|\mathcal{G}) = \prod_{i=1}^{n} P(X_i|\mathbf{Pa}_i, \mathcal{G})$$

Arooshi Avasthy arooshia@buffalo.edu

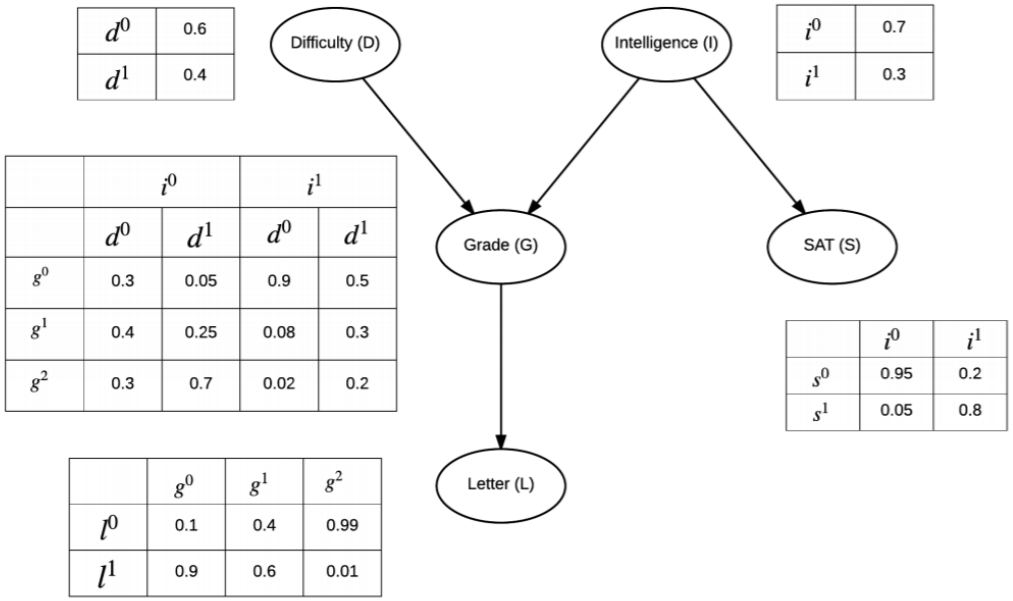

Fig. 1: *Student Model: A simple Bayesian Network.*

Markov Network: A Markov Network consists of an undirected graph and a few Factors are associated with it. Unlike Conditional Probability Distributions, a Factor does not represent the probabilities of variables in the network; instead it represents the compatibility between random variables that is how much a particular state of a random variable likely to agree with the another state of some other random variable.

Creating Markov Models in pgmpy A Markov Network consists of an undirected graph which connects the random variables according to the relation between them. A markov network is parameterized by factors which represent the likelihood of a state of one variable to agree with some state of other variable.

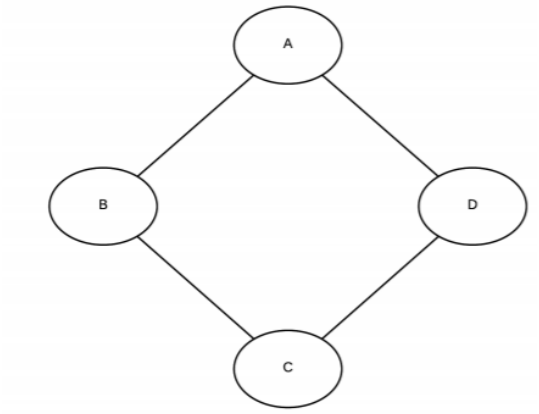

Fig. 2: *A simple Markov Model*

Tasks Implemented:

Task 1:

Evaluate pairwise correlations and independences that exist in the data.

For the multi-categorical variables, following methods can measure the closeness of P(x,y) and P(x)P(y):

- Calculating the cross entropy between P(x, y) and P(x)P(y)
- Calculating ∑abs ((P(x,y) - P(x)P(y)))
- Calculating ∑abs ((P(x|y) - P(x))) for some values of y with large P(y), and then ∑abs ((P(y|x) - P(y))) for some values of x with large P(x).

For our experiment correlation between x1,x2,x3,x4,x5,x6 nodes is calculated using ∑abs ((P(x,y) - P(x)P(y))).

Task 2:

Model Creation:

- Created three bayesian models by adding nodes x1-x6 and adding relations between nodes based intuition and based on threshold value.
- Correlations with values more than 0.2 were taken into consideration and CPD corresponding to the relation is added to the model.

```
model1 = BayesianModel()
model1.add_nodes_from(['x1', 'x2', 'x3','x4','x5','x6'])
model1.add_edges_from([('x1', 'x4'), ('x4', 'x2'),('x2','x5'),('x5','x3'),('x3','x6')])
x1_cpd=TabularCPD('x1',4,[[0.780],[0.015],[0.055],[0.15]])
model1.add_cpds(x4x1_cpd,x2x4_cpd,x5x2_cpd,x3x5_cpd,x6x3_cpd,x1_cpd)
```

Data Generation:

Sampling from directed graphical models

Sampling methods can be used to perform both marginal and MAP inference queries; in addition, they can compute various interesting quantities, such as expectations E[f(X)] of random variables distributed according to a given probabilistic model. Sampling methods have historically been the main way of performing approximate inference.

Our technique for sampling from multinomials naturally extends to Bayesian networks with multinomial variables, via a method called ancestral (or forward) sampling. Given a probability p(x1,…,xn)p(x1,…,xn)specified by a Bayes net, we sample variables in topological order. We start by sampling the variables with no parents; then we sample from the next generation by conditioning these variables' CPDs to values sampled at the first step. We proceed like this until all n variables have been sampled. Importantly, in a Bayesian network over n variables, forward sampling allows us to sample from the joint distribution x∼p(x) in linear O(n) time by taking exactly 1 multinomial sample from each CPD.

```
from pgmpy.sampling import BayesianModelSampling

inference1 = BayesianModelSampling(model1)

dataModel1= np.array(inference1.forward_sample(size=1000, return_type='recarray'))
```

Implementing Hill Climb Search and using K2 score over the dataset to find the best model

- We experimentally study the K2 algorithm in learning a Bayesian network (BN) classifier. Starting from an initial BN structure, the K2 algorithm searches the BN structure space and selects the structure maximizing the K2 metric. The K2 uses a greedy search and may impose no restriction on the number of parents a node has. The K2 search begins by assuming that a node has no parents and then adds incrementally that parent from a given ordering whose addition increases the score of the resulting structure the most. We stop adding parents to the node when the score stops to increase. A common scoring metric is the Bayesian score that is, in principle, the posterior probability of a structure G given a random sample D

$$P(\mathcal{G}|D) = \frac{P(D|\mathcal{G})P(\mathcal{G})}{P(D)} = \frac{P(\mathcal{G}, D)}{P(D)}$$

- Hill Climb Search is for BayesianModels, is used to learn network structure from data and attempts to find a model with optimal score.

```
from pgmpy.estimators import HillClimbSearch

hc = HillClimbSearch(data, scoring_method=K2Score(data))

best_model = hc.estimate()

print(best_model.edges())
```

```
[('x1', 'x2'), ('x6', 'x5'), ('x6', 'x3'), ('x2', 'x6'), ('x2', 'x4'),
('x3', 'x5'), ('x4', 'x5'), ('x4', 'x3'), ('x4', 'x6')]
```

Task 3:

Bayesian to Markov Model

The markov model created would be the moral graph of the bayesian model.

```
def to_markov_model(self):
    moral_graph = self.moralize()
    mm = MarkovModel(moral_graph.edges())
    mm.add_factors(*[cpd.to_factor() for cpd in self.cpds])

    return mm
mm = bayesianmodel.to_markov_model()
```

Moralized Graph

• Moral graph M(G) of a Bayesian network G is an undirected graph. It contains an undirected edge between X and Y if: - There is a directed edge between them in the BN OR

- X and Y are both parents of the same node

It follows that

1. M(G) is a minimal I-map for G
2. If G is moral then M(G) is a perfect map of G

Inference Task: P(Y|E=e)

– Posterior probability distribution over values y of Y
– Conditioned on the fact E=e
– Can be viewed as Marginal over Y in distribution we obtain by conditioning on e

Inference in BNs is the task of calculating the conditional probability distribution of a subset of the nodes in the graph (the ‘‘hidden’’ nodes2) given another subset of the nodes (the ‘‘observed’’ nodes).

```
inference_mm = VariableElimination(mm)
comp = inference.query(['x2'])['x2']
print(comp)
```

Task 4:

And Dataset

• Convert the dataset into dataframe format and use Hill Climb Search to find the best model through K2 score.
• Create the model based on node relations obtained through Hill Climb Search.
• Use BayesianEstimator to estimate CPDs for the given model.

```
from pgmpy.estimators import BayesianEstimator
est = BayesianEstimator(model, andData2)

f1_cpd=est.estimate_cpd('f1', prior_type='K2', equivalent_sample_size=20)
```