Project 1: Determining Probabilities of Handwriting Formations using PGMs

Ravi Teja Sunkara
Department of Computer Science
University at Buffalo, NY 14214
rsunkara@buffalo.edu

Abstract

Probabilistic graphical models (PGMs) have been developed to determine the probabilities of observations for handwriting patterns which are described by several variables obtained by document examiners. Multiple PGMs were built for handwritten 'th' and 'and' image datasets and comparisons were provided.

1. PGMs for 'th' dataset

1.1 Dependencies between the variables

Each 'th' image dataset observation has 6 variables describing it. For a PGM to be constructed, various dependencies between the variables need to be determined. These dependencies were determined using the formula $\sum abs((P(x,y) - P(x)P(y))$. If a value obtained (colloquially referred to as 'correlation') between two variables is less than 1.14 (threshold) then they were considered to be independent.

variable	x1	x2	x3	x4	x5	x6
x1	1.0000	0.1598	0.0000	0.1194	0.0000	0.1602
x2	0.0000	1.0000	0.2185	0.0000	0.0947	0.0000
x3	0.0000	0.2346	1.0000	0.0000	0.1128	0.1192
x4	0.1196	0.1157	0.0000	1.0000	0.0000	0.1435
x5	0.0000	0.8528	0.1178	0.0000	0.0000	0.0000
x6	0.1768	0.1753	0.1390	0.1431	0.0000	0.0000

Figure 1: Correlations between variables

1.2 Building Bayesian Models

Bayesian models were constructed randomly by hand and edges were defined if the two variables had a correlation value above the set threshold [1] (all the nodes were used to create a Bayesian model). A total of 5 different random models were defined in this way. For each model based on the edges defined, CPDs were created using TabularCPD and then they were normalized.
A bayesian model's nodes, edges and CPDs were defined using add_nodes_from, add_edges_from and add_cpds attributes of the model.

[1] No V-structures were taken into consideration while building the models due to the mathematical complexity of defining the CPDs.

```
# Second Model
model2 = BayesianModel()
model2.add_nodes_from(['x1','x2','x3','x4','x5','x6'])
model2.add_edges_from([('x1','x2'),('x1','x4'),('x6','x1'),('x2','x3'),('x2','x5')])
model2.add_cpds(cpd_x6,cpd_x1x2,cpd_x1x4,cpd_x6x1,cpd_x2x3,cpd_x2x5)
```

Figure 2: An example of a defined Bayesian Model

1.3 Evaluating the models using K2 Score

1.3.1 On their own generated data

From each of the defined model, a dataset of 1000 observations was generated using ancestral sampling. The K2 Scores have been obtained for each of the model on the generated datasets. Model 1 is the best model among the five on their own generated datasets.

Table 1: K2 Scores of Bayesian Models on their own generated data

Model	K2 Scores
Model 1	-6389.48
Model 2	-6399.85
Model 3	-6447.47
Model 4	-6446.3
Model 5	-6448.48

1.3.2 On the entire dataset

The data obtained from the 5 different models has been combined which created a dataset of 50,000 observations. K2 scores of each individual model has been calculated on this dataset. Model 1 performed the best in this scenario as well.

Table 2: K2 Scores of the Bayesian Models on the entire dataset of 50,000 observations

Model	K2 Scores
Model 1	-32029.7
Model 2	-32036.1
Model 3	-32128.4
Model 4	-32126.6
Model 5	-32433.7

1.4 High and Low probability pattern of ‘th’

From the dataset generated by the best model (model1 in here), the most repeated patterns for high and low probabilities has been determined using groupby and count functionalities.

The high probability has a pattern of 011031 and occurs 21 times. Please see below.

	x1	x2	x3	x4	x5	x6	count
106	0	1	1	0	3	1	21
108	0	1	1	0	3	3	19
16	0	0	1	0	0	3	18

The low probability has a pattern of 000001 and occurs 1 time. Please see below.

	x1	x2	x3	x4	x5	x6	count
0	0	0	0	0	0	1	1
302	2	0	1	0	2	3	1
300	2	0	1	0	1	1	1

Figure 3: Patterns for high and low probability 'th'

A high probability 'th' has – t shorter than h, with loop of h curved right side and straight left side, a pointed h, with the cross on t on upper half of staff, with no set pattern for baseline of h and a single stroke/closed t.
A low probability 'th' has – t taller than h, with the loop of h having a retraced shape, a pointed arch of h, with the cross on t on upper half of staff, with an even baseline of h, closed/single stroke of t.

1.5 Additional Task

Hill climb search was performed on the dataset generated from all the models (50,000 observations) to find the best Bayesian model edges. A Bayesian model has been created using the edges and then Bayesian parameter estimation has been used to determine the CPDs for this model. The model thus created had the best K2 Score (-32153.26) when compared with all the models above.

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

Figure 4: The edges of the best Bayesian model obtained through Hill climb search on the entire dataset of 50,000 observations obtained by ancestral sampling.

1.6 Markov Network

So far, we have only looked at Bayesian network which is a directed graph. Another type of PGM in which the underlying graph is undirected is called Markov network. The best Bayesian Model obtained above using Hill Climb Search was converted to a Markov network using to_markov.model()

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

Figure 5: The edges of the Markov Network

1.7 Inference comparison between Bayesian and Markov Network

The inferences were obtained using Variable Elimination method. The inferences for both Bayesian and Markov are same.

```
Inferences using Bayesian Network:
Probability of x2:
+------+-----------+
| x2   |   phi(x2) |
+======+===========+
| x2_0 |    0.2826 |
+------+-----------+
| x2_1 |    0.3152 |
+------+-----------+
| x2_2 |    0.1096 |
+------+-----------+
| x2_3 |    0.1136 |
+------+-----------+
| x2_4 |    0.1791 |
+------+-----------+

Coditional Probabilty of x5 given x2=1, x3=1, x1=0, x4=0, x6=1:
+------+-----------+
| x5   |   phi(x5) |
+======+===========+
| x5_0 |    0.3357 |
+------+-----------+
| x5_1 |    0.1100 |
+------+-----------+
| x5_2 |    0.1290 |
+------+-----------+
| x5_3 |    0.4254 |
+------+-----------+
```

Figure 7: Inferences obtained using Bayesian Model

```
Inferences using Markov network:
Probability of x2:
+------+-----------+
| x2   |   phi(x2) |
+======+===========+
| x2_0 |    0.2826 |
+------+-----------+
| x2_1 |    0.3152 |
+------+-----------+
| x2_2 |    0.1096 |
+------+-----------+
| x2_3 |    0.1136 |
+------+-----------+
| x2_4 |    0.1791 |
+------+-----------+

Coditional Probabilty of x5 given x2=1, x3=1, x1=0, x4=0, x6=1:
+------+-----------+
| x5   |   phi(x5) |
+======+===========+
| x5_0 |    0.3357 |
+------+-----------+
| x5_1 |    0.1100 |
+------+-----------+
| x5_2 |    0.1290 |
+------+-----------+
| x5_3 |    0.4254 |
+------+-----------+
```

Figure 8: Inferences using Markov Network

If one looks at the data keenly, the values of other variables I used for obtaining the conditional probability of x5 match with the high probability pattern of 'th'. In high probability pattern of 'th' when the other values of variables are fixed, x5 takes 3. The same can be inferred from above as the probability of x5 taking value 3 is highest with 0.4254

1.8 **Computation time comparison between Bayesian and Markov**

Computation times were obtained using time.clock(), empirically Markov is taking less time when compared to Bayesian network.

```
# Computational time of inference using Bayesian Network
import time
time_start = time.clock()
infer = VariableElimination(model)
infer.query(['x3'])['x3']
infer.query(['x2'], evidence={'x1':0, 'x6':1}) ['x2']
time_elapsed = (time.clock() - time_start)
print('Computation time for inference using "Bayesian Network": ', time_elapsed)
```

```
Computation time for inference using "Bayesian Network":  0.04466309999997975
```

```
# Computational time of inference using Markov Network
import time
time_start = time.clock()
infer = VariableElimination(mm)
infer.query(['x3'])['x3']
infer.query(['x2'], evidence={'x1':0, 'x6':1}) ['x2']
time_elapsed = (time.clock() - time_start)
print('Computation time for inference using "Markov Network": ', time_elapsed)
```

```
Computation time for inference using "Markov Network":  0.034379699999988134
```

Figure 9: Computational time comparison of Bayesian and Markov networks

2. PGMs for ‘and’ dataset

2.1 Building Bayesian Models

A searching strategy called, Hill Climb search has been performed to obtain the best Bayesian model that is likely to fit the data.

[('f3', 'f4'), ('f3', 'f9'), ('f3', 'f8'), ('f5', 'f9'), ('f5', 'f3'), ('f9', 'f8'), ('f9', 'f7'), ('f9', 'f1'), ('f9', 'f6'), ('f9', 'f2'), ('f9', 'f4')]

Figure 10: Edges of the best Bayesian model obtained using Hill Climb search on ‘and’ images dataset

2.2 Evaluating the models using K2 Score

A total of 3 Bayesian models have been created on the ‘and’ image dataset. The goodness score or to measure the fit between model and the data, K2 Score has been calculated for each of these models. The model 1 (which was obtained using Hill Climb Search) gave the best K2 score.

```
Model 1 K2 Score: -9462.704892371386
Model 2 K2 Score: -9597.068195873735
Model 3 K2 Score: -9463.727521719544
```

Figure 11: K2 scores comparison of different models on the ‘and’ image dataset

2.3 Inferences comparison between Bayesian and Markov Network

The inferences were obtained using Variable Elimination method and the inferences are the same for both Bayesian and Markov networks.

```
Inferences using Bayesian Network-
Probability of f2:
+------+-----------+
| f2   |   phi(f2) |
+======+===========+
| f2_0 |    0.1775 |
+------+-----------+
| f2_1 |    0.5138 |
+------+-----------+
| f2_2 |    0.1313 |
+------+-----------+
| f2_3 |    0.0135 |
+------+-----------+
| f2_4 |    0.1638 |
+------+-----------+

Probability f5 given f4=0 and f1=1:
+------+-----------+
| f5   |   phi(f5) |
+======+===========+
| f5_0 |    0.1921 |
+------+-----------+
| f5_1 |    0.0374 |
+------+-----------+
| f5_2 |    0.7614 |
+------+-----------+
| f5_3 |    0.0091 |
+------+-----------+
```

Figure 12: Inferences using Bayesian Network

```
Inferences using Markov Network-
Probability of f2:
+------+-----------+
| f2   |   phi(f2) |
+======+===========+
| f2_0 |    0.1775 |
+------+-----------+
| f2_1 |    0.5138 |
+------+-----------+
| f2_2 |    0.1313 |
+------+-----------+
| f2_3 |    0.0135 |
+------+-----------+
| f2_4 |    0.1638 |
+------+-----------+

Probability f5 given f4=0, f2=1, f3=1, f1=0, f6=1:
+------+-----------+
| f5   |   phi(f5) |
+======+===========+
| f5_0 |    0.2424 |
+------+-----------+
| f5_1 |    0.0293 |
+------+-----------+
| f5_2 |    0.7215 |
+------+-----------+
| f5_3 |    0.0068 |
+------+-----------+
```

Figure 13: Inferences obtained using Markov Network

2.4 Computation time comparison between Bayesian and Markov Network

Computation times were obtained using time.clock(), empirically Markov is taking less time when compared to Bayesian network.

```
# Computational time of inference using Bayesian Network
import time
time_start = time.clock()
infer = VariableElimination(model1)
infer.query(['f3'])['f3']
infer.query(['f2'], evidence={'f1':0, 'f6':1}) ['f2']
time_elapsed = (time.clock() - time_start)
print('Computation time for inference using "Bayesian Network": ', time_elapsed)
```

```
Computation time for inference using "Bayesian Network":  0.04413220001151785
```

```
# Computational time of inference using Bayesian Network
import time
time_start = time.clock()
infer = VariableElimination(mm1)
infer.query(['f3'])['f3']
infer.query(['f2'], evidence={'f1':0, 'f6':1}) ['f2']
time_elapsed = (time.clock() - time_start)
print('Computation time for inference using "Markov Network": ', time_elapsed)
```

```
Computation time for inference using "Markov Network":  0.037094400002388284
```

Figure 14: Computation times of Bayesian and Markov networks