Determining Probabilities of Handwriting Formations using PGMs

Shruti Bendale
Person Number: 50289048,
School of Engineering and Applied Sciences,
University at Buffalo, Buffalo, NY, 14214
shrutitu@buffalo.edu

Abstract

The purpose of this project is to determine whether a particular handwriting sample is common or rare. We work with handwriting patterns which are described by document examiners. The document examiners analyzed handwriting samples of 'th' based on different features. Probabilistic graphical models (Bayesian and Markov models) were created using the Conditional Probability Distribution tables generated from the analysis of the examiners and the inferences were compared. Furthermore, we use the 'and' dataset to generate PGMs and the Conditional Probability Distribution tables.

1. Introduction

The problem of handwriting identification is to interpret intelligible handwritten input automatically, which is of great interest in the pattern recognition research community because of its applications to many fields. As one of the fundament problems in designing practical recognition systems, the recognition of handwritten digits is an active research field. Immediate applications of the digit recognition techniques include postal mail sorting, automatic address reading and mail routing, bank check processing, etc.

In this project we were given data for 200 handwriting sample compiled according to percentage values. Numerical and percentage values for correlations between handwriting characteristics and the writers were given in 7 tables. The data was cleaned and converted into decimal format to generate the conditional probability tables for all the variables. We then evaluate the correlation and independencies between variables by calculating the cross entropy. We further generate some Bayesian models using these dependencies and evaluate the best model by calculating the K2 scores. Using the best model, we describe the high probability and low probability formation of '*th*'. This model is then converted into a Markov model to get similar inferences. We are also given data for the formation of '*and*' and we have to construct a Bayesian model.

2. Implementation

Cleaning the probability tables:

The percentages were converted into a decimal format and the values and features were further cleaned to give the marginal and conditional probabilities. The marginal probabilities were stored in the p(x1), p(x2), p(x3), p(x4), p(x5), p(x6) dataframes(Figure 2.0.1) and the conditional probabilities were stored in (x2|x1), p(x4|x1), p(x6|x1), p(x3|x2)p(x5|x2), p(x2|x3), p(x5|x3), p(x6|x3), p(x1|x4), p(x2|x4), p(x6|x4), p(x2|x5), p(x3|x5), p(x1|x6), p(x2|x6), p(x3|x6) and p(x4|x6) dataframes(Figure 2.0.2)

	x1
0	0.780
1	0.015
2	0.055
3	0.150

Figure 2.0.1: Example for a marginal *Probability dataframe(x1)*

	x01	x11	x21	x31
0	0.231	0.666	0.455	0.400
1	0.365	0.000	0.091	0.200
2	0.026	0.000	0.000	0.033
3	0.173	0.000	0.182	0.167
4	0.205	0.333	0.273	0.200

Figure 2.0.2: *Example for a conditional probability dataframe (x2|x1)*

Task 1: Evaluating correlations and independences

We determine the dependencies between two features based on the intuition that,
if $p(x, y) = p(x)\, p(y)$, x and y are independent
So, $p(x, y) - p(x)\, p(y)$ should be 0 (or close to 0)
So, $p(x|y)\, p(y) - p(x)\, p(y)$ should be 0 (or close to 0)

The code snippet for calculating the entropy for x1 and x2 is as follows:

```
for i in range(0,4):
    for j in range(0,5):
        closeness += abs( (x2_x1[j][i] * x1[i]) - (x1[i]*x2[j]))   #p(x2,x1) = p(x2|x1) p(x1)
print("Entropy for x2|x1 is: ", closeness[0] )
```

The cross entropies calculated for all the dependencies is given as follows:

```
Entropy for x2|x1 is:  0.15977
Entropy for x4|x1 is:  0.11943000000000004
Entropy for x6|x1 is:  0.16015500000000005

Entropy for x3|x2 is:  0.21852500000000002
Entropy for x5|x2 is:  0.12926000000000004

Entropy for x2|x3 is:  0.21875800000000006
Entropy for x5|x3 is:  0.11551999999999997
Entropy for x6|x3 is:  0.09498000000000002

Entropy for x1|x4 is:  0.11957000000000002
Entropy for x2|x4 is:  0.53425
Entropy for x6|x4 is:  0.11923999999999993

Entropy for x2|x5 is:  0.13126499999999997
Entropy for x3|x5 is:  0.11596500000000005

Entropy for x1|x6 is:  0.16036999999999996
Entropy for x2|x6 is:  0.17531500000000003
Entropy for x3|x6 is:  0.09434000000000006
Entropy for x4|x6 is:  0.14307000000000003
```

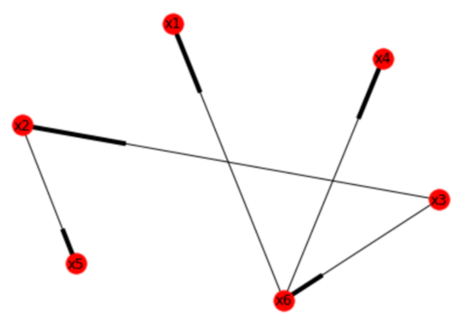
As no entropy is actually 0, we set a threshold value for entropy and consider all the conditional dependencies below the threshold values as 0 (independent). We also assume independence for pairs of variables not appearing in the CPD tables.

```
Let the threshold be 0.14
x 2  depends on  x 1
x 6  depends on  x 1
x 3  depends on  x 2
x 2  depends on  x 3
x 2  depends on  x 4
x 1  depends on  x 6
x 2  depends on  x 6
x 4  depends on  x 6
```

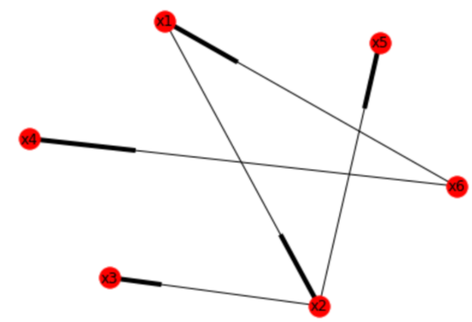
Task 2: Constructing Bayesian network

We construct five Bayesian models based on the dependencies generated above and calculate the K2 scores of all the models using data generated by Bayesian model sampling. Bayesian sampling takes into consideration the CPDs for the model and generates a sample of data which is used to calculate the K2 score. We generate different data for all models and compare the K2 scores.

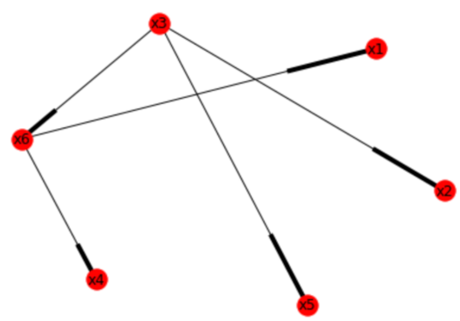
The five models were which were considered are:



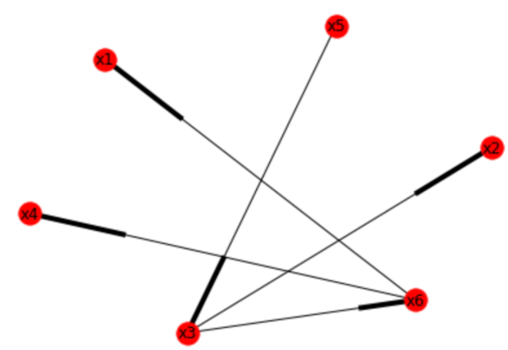
Model 1 K2 Score: -31692.129247956276



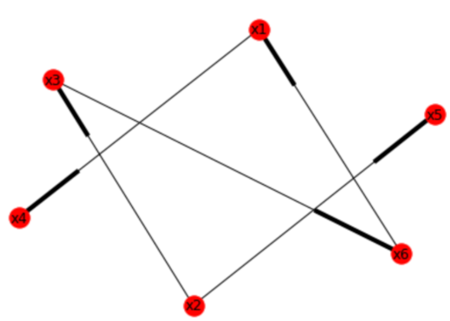
Model 2 K2 Score: -6397.185138400006



Model 3 K2 Score: -6462.709488213288



Model 4 K2 Score: -6445.163742235411



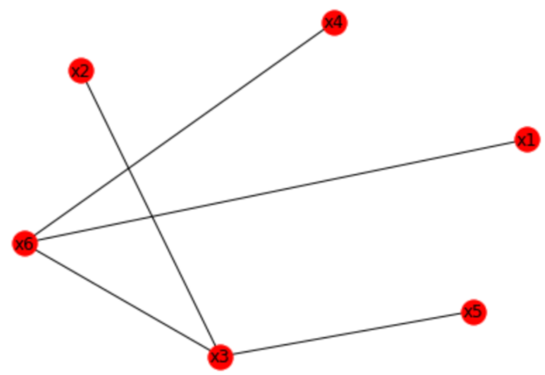
Model 5 K2 Score: -6507.860253229551

Model 2 was the best Bayesian model as it had the highest K2 score.

Task 3: Converting the best Bayesian model to Markov model

The best Bayesian model (model 2) was converted into Markov model with undirected edges using moralization. We use Gibbs sampling to sample data for the Markov network. The resulting Markov network is as follows:

```
The markov model is:
[('x5', 'x3'), ('x3', 'x2'), ('x3', 'x6'), ('x6', 'x4'), ('x6', 'x1')]
```



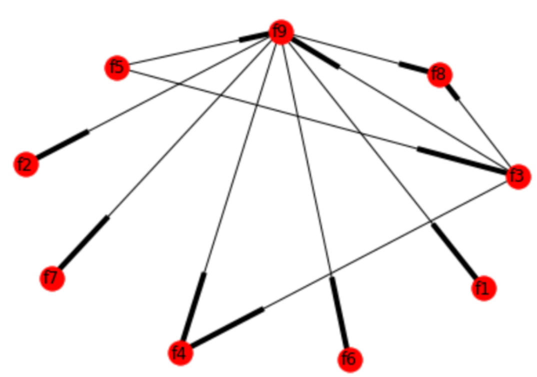
Task 4: Bayesian and Markov network using the 'and' dataset

We use hill climbing search to search for the best Bayesian model for the given ‘and’ dataset. The code snippet is as follows:

```
hc = HillClimbSearch(and_data, scoring_method=K2Score(and_data))
best_model = hc.estimate()
and_model = BayesianModel(best_model.edges())
```

The best Bayesian model generated using hill climbing search is as follows:

```
The best bayesian network is:
```




```
K2 Score: -9462.704892371386
```

We then generate the CPDs for all the nodes in this model using the maximum likelihood estimator. An example for the CPD of node f1 is given as follows:

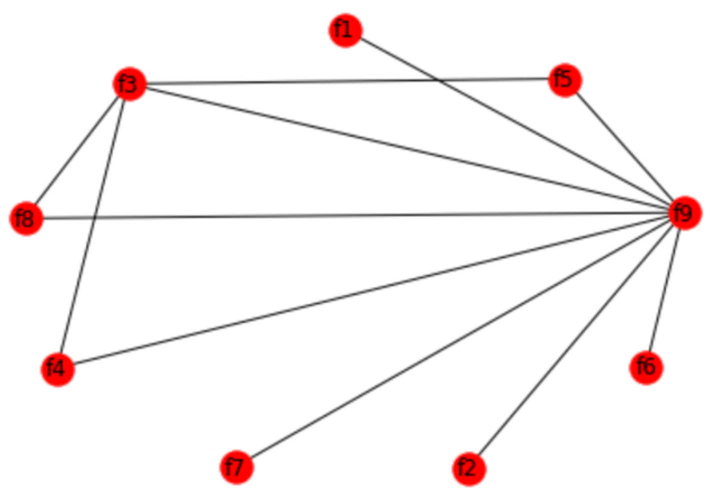
```
cpds = MaximumLikelihoodEstimator(and_model, and_data)
print("\n\nCPD table for f1: \n", cpds.estimate_cpd('f1'))
```

CPD table for f1:

f9	f9(1)	f9(2)
f1(0)	0.19823788546255505	0.10638297872340426
f1(1)	0.44933920704845814	0.26032540675844806
f1(2)	0.2643171806167401	0.3692115143929912
f1(3)	0.0881057268722467	0.2640801001251564

We also convert the above Bayesian network to a Markov Model given as follows:

The markov network is:



3. Inferences

After determining the best Bayesian model for the ‘th’ dataset and converting it to Markov model, we have to see which formations of ‘th’ has the highest and the lowest probabilities of repeating for both of the models. For this, we look into the data generated for both of these models, drop the duplicate formations and then group samples by patterns.

The inferences for the Bayesian Network are:

high probability th is : x5 0 x3 1 x2 0 x6 3 x4 0 x1 0 count 20	**Inferences for high probability ''th'':** Baseline of the "h": baseline slanting upward Shape of the arch of the "h": pointed arch Shape of the loop of the "h": retraced staff Shape of the ''t'': single stroked ''t'' Height of cross on "t" staff: cross in upper half Height relationship of "t" to "h": ''t'' shorter
low probability th is : x5 0 x3 0 x2 0 x6 1 x4 0 x1 2 count 1	**Inferences for low probability ''th'':** Baseline of the "h": baseline slanting upward Shape of the arch of the "h": rounded arch Shape of the loop of the "h": retraced staff Shape of the ''t'': made closed ''t'' Height of cross on "t" staff: cross in upper half Height relationship of "t" to "h": ''t'' taller

The inferences for the Markov network are:

high probability th is : x5 0 x3 1 x2 0 x6 3 x4 0 x1 0 count 20	**Inferences for high probability ''th'':** Baseline of the "h": baseline slanting upward Shape of the arch of the "h": pointed arch Shape of the loop of the "h": retraced staff Shape of the ''t'': single stroked ''t'' Height of cross on "t" staff: cross in upper half Height relationship of "t" to "h": ''t'' shorter
low probability th is : x5 0 x3 0 x2 0 x6 1 x4 0 x1 2 count 1	**Inferences for low probability ''th'':** Baseline of the "h": baseline slanting upward Shape of the arch of the "h": rounded arch Shape of the loop of the "h": retraced staff Shape of the ''t'': made closed ''t'' Height of cross on "t" staff: cross in upper half Height relationship of "t" to "h": ''t'' taller

4. Conclusion

We calculated the K2 scores for all the Bayesian models generated and picked the one with the best K2 score. We then converted this model to a Markov model and compared the inferences. Looking at the inferences we can say that the Bayesian network and the Markov network came with the same inferences for high probability of "th" and low probability of ''th''.