UNIVERSITY OF CAMBRIDGE

Latent Gaussian Processes for Distribution Estimation of Multivariate Categorical Data

Yarin Gal • Yutian Chen • Zoubin Ghahramani

yg279@cam.ac.uk

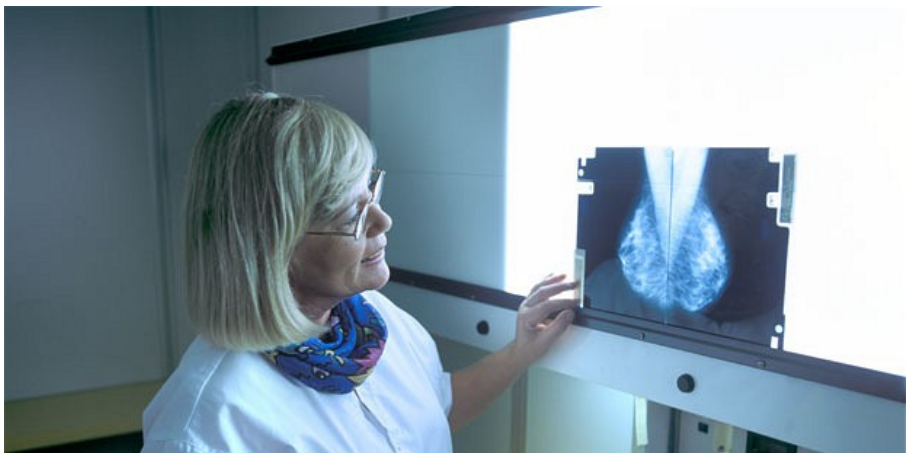
Multivariate Categorical Data

Multivariate Categorical Data

UNIVERSITY OF CAMBRIDGE

Multivariate Categorical Data

UNIVERSITY OF CAMBRIDGE



Distribution Estimation

Given

$$D = \{\mathbf{y}_n = (y_{n1}, ..., y_{nD}) \mid n = 1, ..., N\}$$

with

$$y_{nd} \in \{1, ..., K\},$$

we want to **learn a distribution**

$$P(\mathbf{y})$$

from D.

Example: Wisconsin Breast Cancer

UNIVERSITY OF CAMBRIDGE

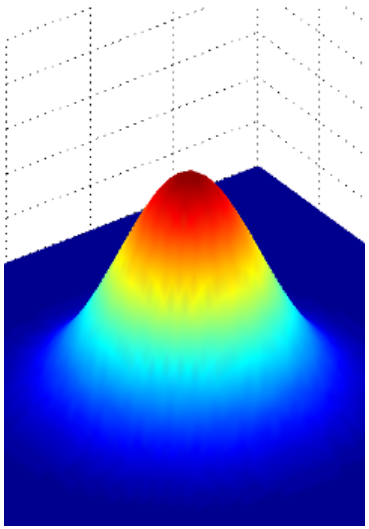
Sample code	Thickness	Unif. Cell Size	Unif. Cell Shape	Marginal Adhesion	Epithelial Cell Size	Bare Nuclei	Bland Chromatin	Normal Nucleoli	Mitoses	Class
1000025	5	1	1	1	2	1	3	1	1	Benign
1002945	5	4	4	5	7	10	3	2	1	Benign
1015425	3	1	1	1	2	2	3	1	1	Benign
1016277	6	8	8	1	3	4	3	7	1	Benign
1017023	4	1	1	3	2	1	3	1	1	Benign
1017122	8	10	10	8	7	10	9	7	1	Malignant
1018099	1	1	1	1	2	10	3	1	1	Benign
1018561	2	1	2	1	2	1	3	1	1	Benign
1033078	2	1	1	1	2	1	1	1	5	Benign
1033078	4	2	1	1	2	1	2	1	1	Benign
1035283	1	1	1	1	1	1	3	1	1	Benign
1036172	2	1	1	1	2	1	2	1	1	Benign
1041801	5	3	3	3	2	3	4	4	1	Malignant
1043999	1	1	1	1	2	3	3	1	1	Benign
1044572	8	7	5	10	7	9	5	5	4	Malignant
1047630	7	4	6	4	6	1	4	3	1	Malignant
1048672	4	1	1	1	2	1	2	1	1	Benign
1049815	4	1	1	1	2	1	3	1	1	Benign
1050670	10	7	7	6	4	10	4	1	2	Malignant
1050718	6	1	1	1	2	1	3	1	1	Benign
1054590	7	3	2	10	5	10	5	4	4	Malignant
1054593	10	5	5	3	6	7	7	10	1	Malignant
1056784	3	1	1	1	2	1	2	1	1	Benign
1057013	8	4	5	1	2	?	7	3	1	Malignant
1059552	1	1	1	1	2	1	3	1	1	Benign
1065726	5	2	3	4	2	7	3	6	1	Malignant
1066373	3	2	1	1	1	1	2	1	1	Benign
1066979	5	1	1	1	2	1	2	1	1	Benign
1067444	2	1	1	1	2	1	2	1	1	Benign
1070935	1	1	3	1	2	1	1	1	1	Benign
1070935	3	1	1	1	1	1	2	1	1	Benign
1071760	2	1	1	1	2	1	3	1	1	Benign
1072179	10	7	7	3	8	5	7	4	3	Malignant

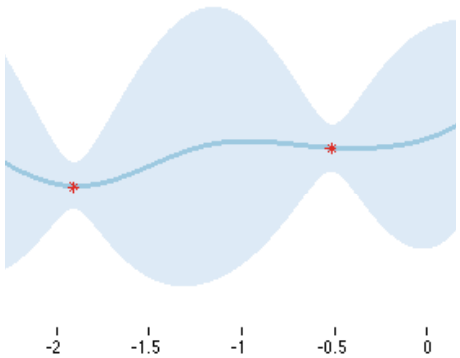
683 patients, 2×10^9 possible configurations

Categorical Latent Gaussian Process

UNIVERSITY OF CAMBRIDGE

We define the generative model as





$$y_{nd} \overset{\text{iid}}{\sim} \text{Softmax}(f_{nd1}, \ldots, f_{ndK})$$

test result

$$\mathbf{y}_n = (y_{n1}, \ldots, y_{nD})$$

medical assessment

Continuous latent space of patients

Dist. over functions generating weights f_{ndk}

N patients $\mathbf{x}_n$, D categorical tests, each with K possible test results

Categorical Latent Gaussian Process

UNIVERSITY OF CAMBRIDGE

1. **Sparse GPs** $\rightarrow$ linear time complexity.
2. **Monte Carlo integration** for the non-conjugate likelihood $\rightarrow$ noisy gradients.
3. **Learning-rate free stochastic optimisation** $\rightarrow$ no fine-tuning of learning-rates.
4. **Symbolic differentiation** $\rightarrow$ simple and modular code

```
import theano.tensor as T
X = m + s * randn(N, Q)
U = mu + L.dot(randn(M, K))
Kmm, Kmn, Knn = RBF(sf2, l, Z), RBF(sf2, l, Z, X), RBFnn(sf2, l, X)
KmmInv = sT.matrix_inverse(Kmm)
A = KmmInv.dot(Kmn)
B = Knn - T.sum(Kmn * KmmInv.dot(Kmn), 0)
F = A.T.dot(U) + B[:,None]**0.5 * randn(N, K)
S = T.nnet.softmax(F)
KL_U, KL_X = get_KL_U(), get_KL_X()
LS = T.sum(T.log(T.sum(Y * S, 1))) - KL_U - KL_X
LS_func = theano.function(['''inputs'''], LS)
dLS_dm = theano.function(['''inputs'''], T.grad(LS, m)) # and others
# ... and run RMS-PROP
```

Categorical Latent Gaussian Process

UNIVERSITY OF CAMBRIDGE

1. **Sparse GPs** $\rightarrow$ linear time complexity.
2. **Monte Carlo integration** for the non-conjugate likelihood $\rightarrow$ noisy gradients.
3. **Learning-rate free stochastic optimisation** $\rightarrow$ no fine-tuning of learning-rates.
4. **Symbolic differentiation** $\rightarrow$ simple and modular code

```
import theano.tensor as T
X = m + s * randn(N, Q)
U = mu + L.dot(randn(M, K))
Kmm, Kmn, Knn = RBF(sf2, l, Z), RBF(sf2, l, Z, X), RBFnn(sf2, l, X)
KmmInv = sT.matrix_inverse(Kmm)
A = KmmInv.dot(Kmn)
B = Knn - T.sum(Kmn * KmmInv.dot(Kmn), 0)
F = A.T.dot(U) + B[:,None]**0.5 * randn(N, K)
S = T.nnet.softmax(F)
KL_U, KL_X = get_KL_U(), get_KL_X()
LS = T.sum(T.log(T.sum(Y * S, 1))) - KL_U - KL_X
LS_func = theano.function(['''inputs'''], LS)
dLS_dm = theano.function(['''inputs'''], T.grad(LS, m)) # and others
# ... and run RMS-PROP
```

Categorical Latent Gaussian Process

UNIVERSITY OF CAMBRIDGE

1. **Sparse GPs** $\rightarrow$ linear time complexity.
2. **Monte Carlo integration** for the non-conjugate likelihood $\rightarrow$ noisy gradients.
3. **Learning-rate free stochastic optimisation** $\rightarrow$ no fine-tuning of learning-rates.
4. **Symbolic differentiation** $\rightarrow$ simple and modular code

```
import theano.tensor as T
X = m + s * randn(N, Q)
U = mu + L.dot(randn(M, K))
Kmm, Kmn, Knn = RBF(sf2, l, Z), RBF(sf2, l, Z, X), RBFnn(sf2, l, X)
KmmInv = sT.matrix_inverse(Kmm)
A = KmmInv.dot(Kmn)
B = Knn - T.sum(Kmn * KmmInv.dot(Kmn), 0)
F = A.T.dot(U) + B[:,None]**0.5 * randn(N, K)
S = T.nnet.softmax(F)
KL_U, KL_X = get_KL_U(), get_KL_X()
LS = T.sum(T.log(T.sum(Y * S, 1))) - KL_U - KL_X
LS_func = theano.function(['''inputs'''], LS)
dLS_dm = theano.function(['''inputs'''], T.grad(LS, m)) # and others
# ... and run RMS-PROP
```

Categorical Latent Gaussian Process

UNIVERSITY OF CAMBRIDGE

1. **Sparse GPs** $\rightarrow$ linear time complexity.
2. **Monte Carlo integration** for the non-conjugate likelihood $\rightarrow$ noisy gradients.
3. **Learning-rate free stochastic optimisation** $\rightarrow$ no fine-tuning of learning-rates.
4. **Symbolic differentiation** $\rightarrow$ simple and modular code

```
import theano.tensor as T
X = m + s * randn(N, Q)
U = mu + L.dot(randn(M, K))
Kmm, Kmn, Knn = RBF(sf2, l, Z), RBF(sf2, l, Z, X), RBFnn(sf2, l, X)
KmmInv = sT.matrix_inverse(Kmm)
A = KmmInv.dot(Kmn)
B = Knn - T.sum(Kmn * KmmInv.dot(Kmn), 0)
F = A.T.dot(U) + B[:,None]**0.5 * randn(N, K)
S = T.nnet.softmax(F)
KL_U, KL_X = get_KL_U(), get_KL_X()
LS = T.sum(T.log(T.sum(Y * S, 1))) - KL_U - KL_X
LS_func = theano.function(['''inputs'''], LS)
dLS_dm = theano.function(['''inputs'''], T.grad(LS, m)) # and others
# ... and run RMS-PROP
```

Back to Breast Cancer...

UNIVERSITY OF CAMBRIDGE

Uniform	(Discrete) Bigram Dirichlet–Multinomial	(Linear) Latent Gaussian Model	**Categorical Latent Gaussian Process**
8.68	3.41	3.57 ± 0.208	$\mathbf{2.86 \pm 0.119}$

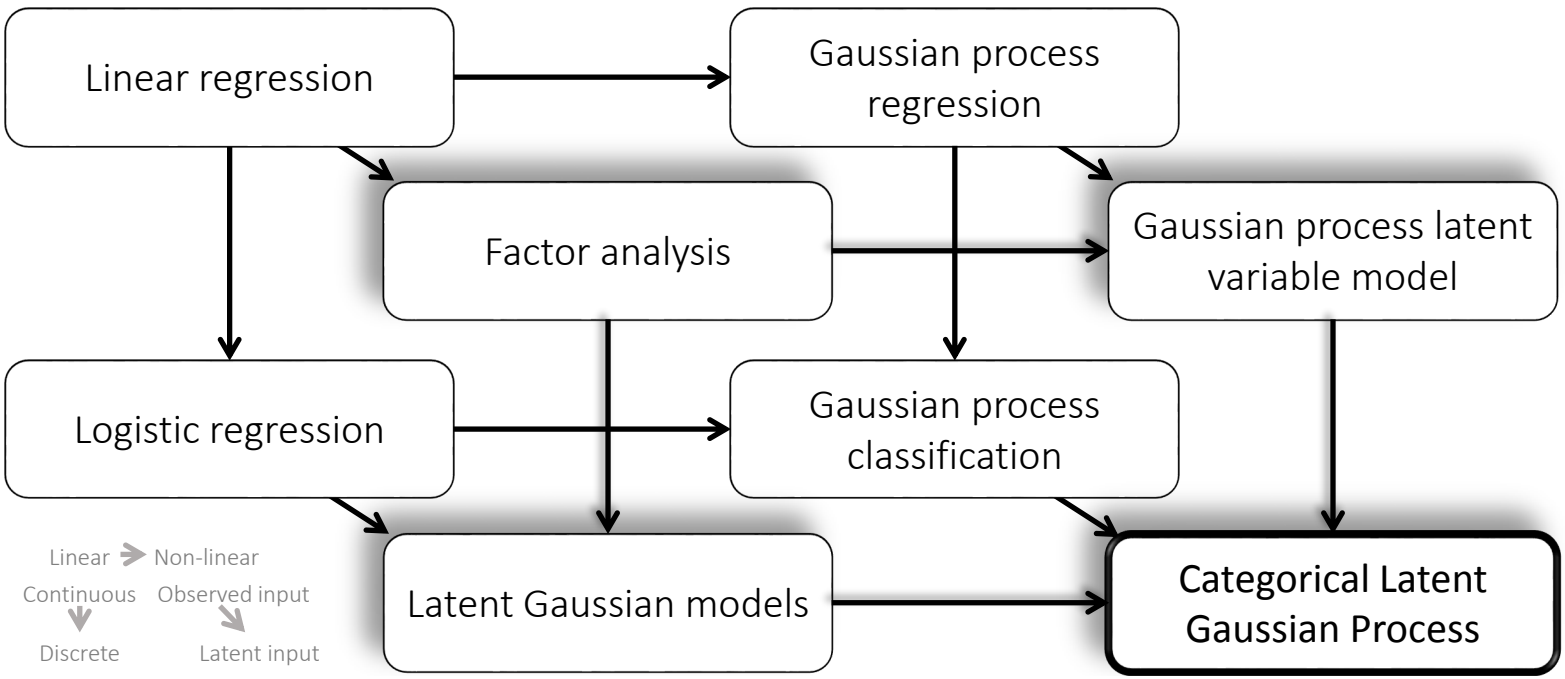
Model perplexity, predicting randomly missing test results

(Lots more results in the poster!)

Thank you

UNIVERSITY OF CAMBRIDGE

Latent Gaussian Processes for Distribution Estimation of Multivariate Categorical Data



Please come and see our poster