Python Package “lcmodels” – Latent Class Models for Evaluation of Classifiers

Eli Samuelson[1, 2], Berkman Sahiner[1], Weijie Chen[1], and Alexej Gossmann[1]

[1]FDA/CDRH/OSEL/DIDSR

[2] The College of Wooster, Wooster, OH

Project

- Implement and investigate Latent Class Models (LCM) for performance assessment of diagnostic tests, human raters, or AI/ML classifiers in absence of the ground truth.
- Create an open source Python package to share with the broad scientific community.
- Gain an in depth understanding about the advantages and the pitfalls of LCM-based techniques for performance assessment without using ground truth data.

Latent Class Model (LCM)

Assuming conditional independence:

$$
\begin{aligned}
&P(X_{i1} = x_{i1}, \ldots, X_{im} = x_{im}) \\
&= P(C_i = 0) \cdot \prod_{j=1,2,\ldots} P(X_{ij} = 1 | C_i = 0)^{x_{ij}} \cdot P(X_{ij} = 0 | C_i = 0)^{x_{ij}} \\
&\quad + P(C_i = 1) \cdot \prod_{j=1,2,\ldots} P(X_{ij} = 1 | C_i = 1)^{x_{ij}} \cdot P(X_{ij} = 0 | C_i = 1)^{x_{ij}},
\end{aligned}
$$

where $x_{i1}, \ldots, x_{im} \in \{0, 1\}$ are each classifier's decisions for patient i, and $C_i \in \{0, 1\}$ is a latent variable representing the true class (ex. diseased or healthy) of patient i.

Modeling dependencies with random effects: Main idea being along the lines of

$$P(X_{ij} = 1 | C_i = 1, T_i = t_i) = \Phi(a_{j1} + b_{j1} t_i),$$

$$P(X_{ij} = 1 | C_i = 1) = \int_{-\infty}^{+\infty} \Phi(a_{j1} + b_{j1} t) \, d\Phi(t) = \Phi\left(\frac{a_{j1}}{\sqrt{1 + b_{j1}^2}}\right) \quad (1)$$

(analogous for $C_i = 0$ using a_{j0} and b_{j0} respectively), where $T_i \sim \mathcal{N}(0, 1)$, Φ is the cumulative distribution function of the standard normal distribution, and a_{jd} and b_{jd} ($j = 1, 2, \ldots, m$ and $d = 0, 1$) are parameters to be estimated.

Computational methods / Optimization: Combines EM algorithm, adaptive Gauss-Hermite quadrature, and BFGS algorithm.

Performance assessment without ground truth: LCM can be used to estimate a classifier's diagnostic sensitivity and specificity without using ground truth data. For example, $\Phi\left(\frac{a_{i1}}{\sqrt{1+b_{i1}^2}}\right)$ in Eq. (1) is an estimate of sensitivity of the m^{th} classifier.[3]

Python Package “lcmodels”

We implement LCM with and without random effects. We begin with a re-implementation of the R package `randomLCA` into Python.[1]

“lcmodels” Python package – model inputs:

- An $n \times m$ matrix of binary decisions of m classifiers on n subjects; may contain missing values.
- Model specification: random effects specification, number of latent classes, etc.

“lcmodels” Python package – model outputs:

- Bayesian information criterion (BIC), log-likelihood, penalized log-likelihood, etc.
- Estimated class probabilities (i.e, prevalence).
- Estimated conditional outcome probabilities for each classifier given true class (i.e., specificity and sensitivity).

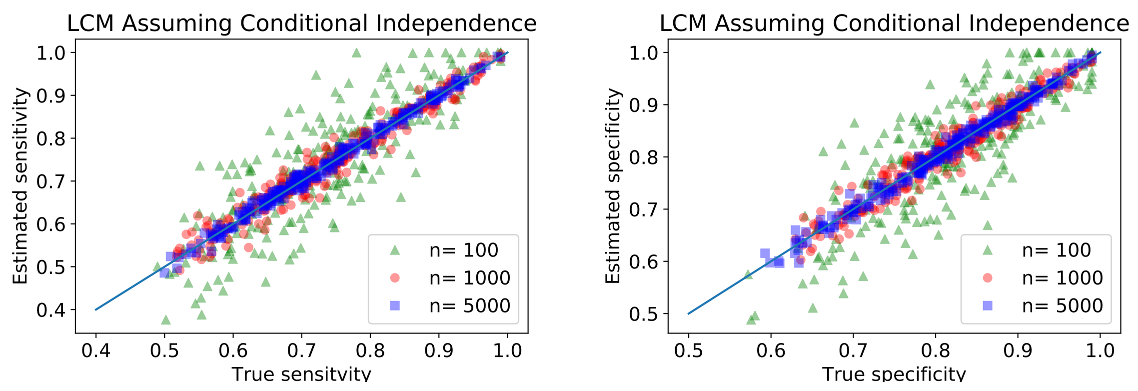
Python Package cont.

- Expected frequency for each pattern of outcomes, which can be compared to the actual observed frequencies.
- Class probabilities for each pattern of outcomes, which can be used to combine classifiers (ensemble learning).
- Confidence intervals and variance estimates for all estimated parameters.

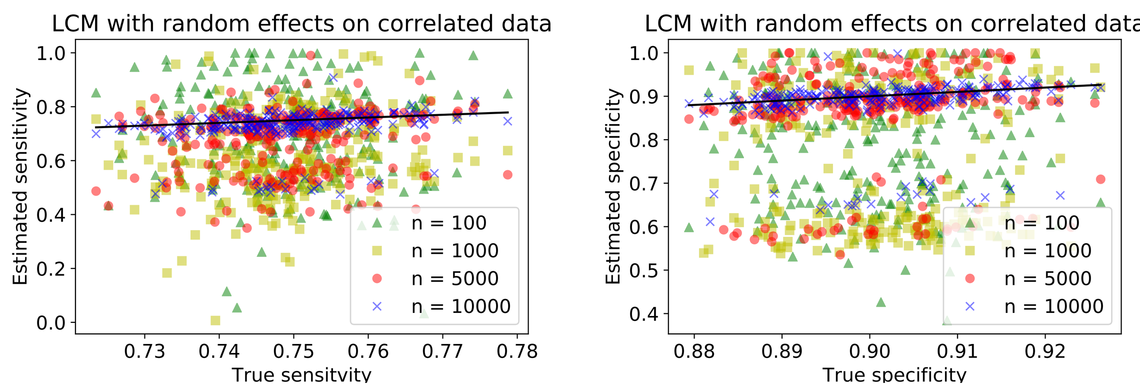
Simulations

Simulated data: Output of 10 Binary classifiers on 100, 1000, 5000, 10000 cases. All simulations were repeated 25 times. For simulation with correlated classifiers the Sensitivity was set at around 75%, and Specificity at around 90%.

Independent classifiers



Correlated classifiers



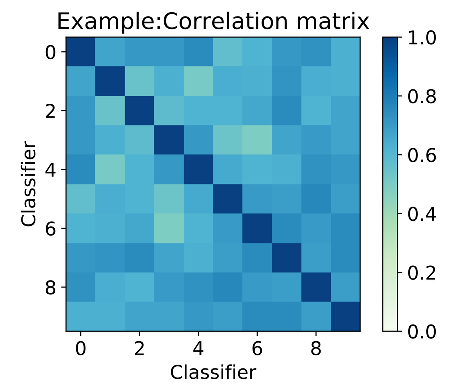


Table: Average Error with Standard Deviation in parentheses

	Independent Classifiers		*Correlated Classifiers*	
# Cases	Sensitivity	Specificity	Sensitivity	Specificity
100	0.0584 (0.0483)	0.0508 (0.0430)	0.1552(0.1131)	0.1383(0.1105)
1000	0.0184 (0.0144)	0.0155 (0.0124)	0.1668(0.1138)	0.1715(0.1285)
5000	0.0080 (0.0060)	0.0075 (0.0062)	0.0908(0.0961)	0.0733(0.0992)
10000	—	—	0.0378(0.0631)	0.0294(0.0682)

Simulated data: Binary scores of 5,10,20,30 classifiers on 130 cases.

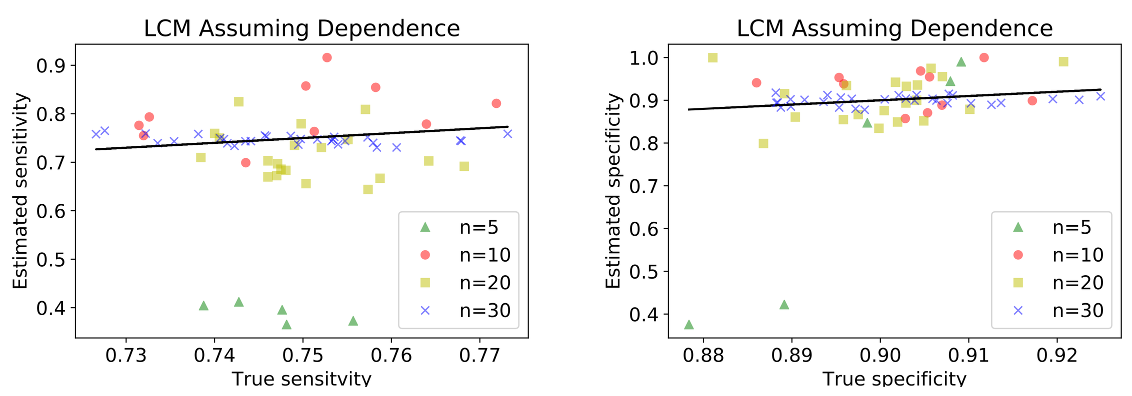


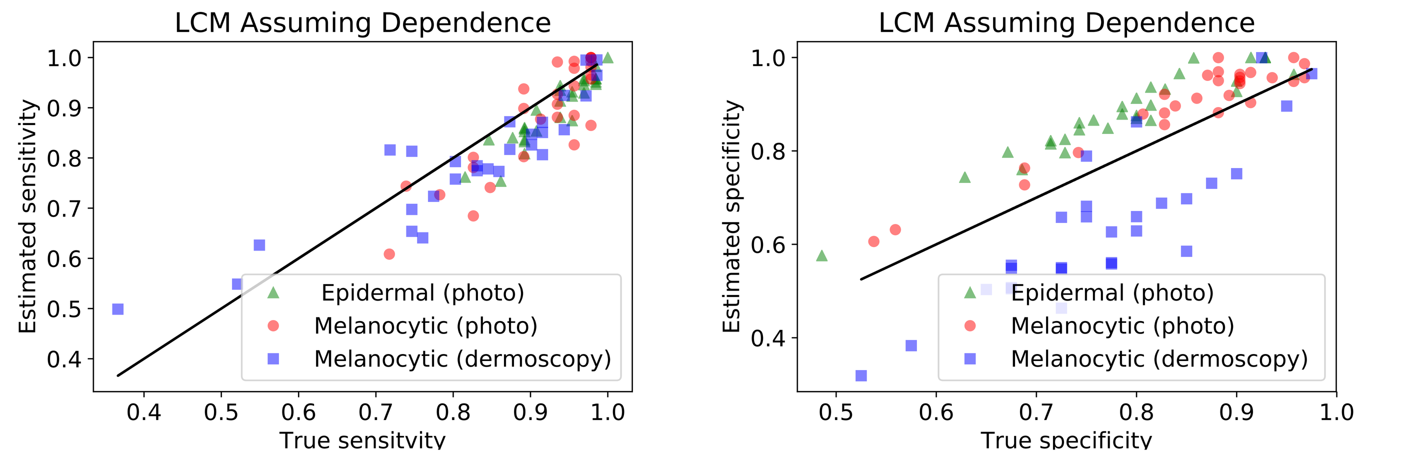
Table: Average Error with Standard Deviation in parentheses

# Readers	Sensitivity	Specificity
5	0.3567 (0.0226)	0.2277 (0.2109)
10	0.0615 (0.0452)	0.0474 (0.0200)
20	0.0536 (0.0452)	0.0451 (0.0263)
30	0.0129 (0.0101)	0.0112 (0.0072)

Results on Real Data

- Esteva et al. tested the performance of their CNN, which classifies skin cancer based on dermoscopic or photographic images, against over 20 dermatologists on over 100 biopsy proven lesion images.[2]

Dataset / Classification task	#Images	#Readers	#CNNs	Prevalence
Epidermal lesions (photographic)	135	26	4	0.481
Melanocytic lesions (photographic)	144	25	4	0.330
Melanocytic lesions (dermoscopic)	111	25	4	0.639



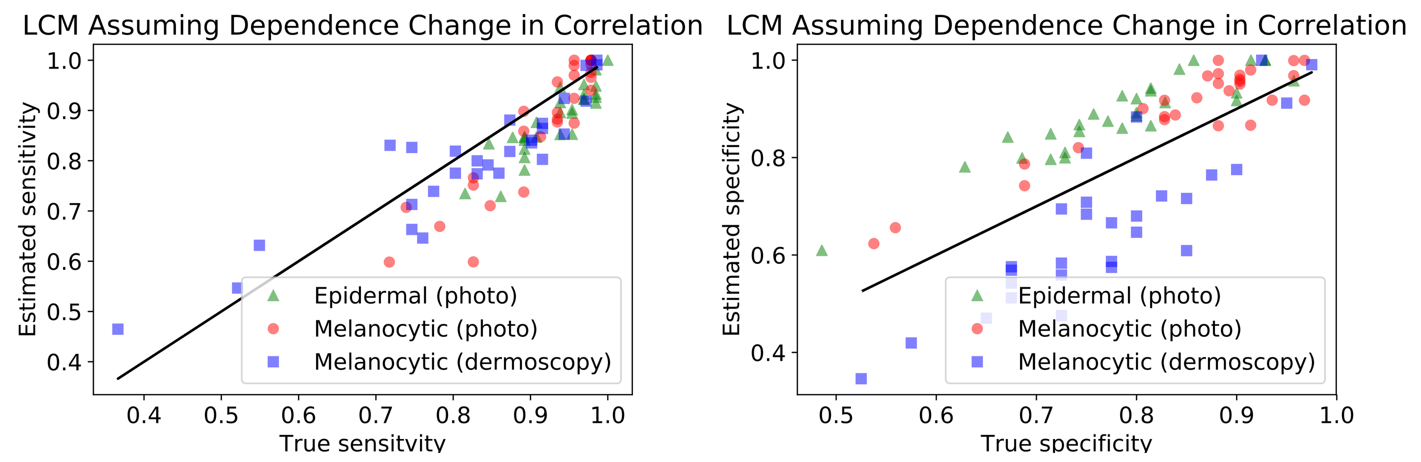


Table: Average Error with Standard Deviation in parentheses

	Assuming Constant Correlation		*Change in Correlation*	
# Cases	Sensitivity	Specificity	Sensitivity	Specificity
Epidermal lesions	0.0354 (0.0255)	0.0889 (0.0298)	0.0517(0.0314)	0.0985(0.0400)
Melanocytic lesions (photographic)	0.0469 (0.0400)	0.0510 (0.0284)	0.0553(0.0513)	0.0621(0.0265)
Melanocytic lesions (dermoscopic)	0.0585 (0.0332)	0.1410 (0.0635)	0.0546(0.0329)	0.1238(0.0595)

Conclusion

- We implemented LCM and LCM with random effects in Python.
- We validated our implementation with simulation studies and on real data.
- If we have independent classifiers then the estimated sensitivity and specificity are highly accurate. If there is dependence between classifiers then LCM does not provide accurate estimates without a very large number of observations.
- We will further optimize the computational efficiency of our software package, and then release it publicly for wider use.

Acknowledgements

We thank Andre Esteva for sharing the data from Ref. [2] with us, and giving us permission to use it for this project.

This project was supported in part by an appointment to the Research Participation Program at the U.S. Food and Drug Administration administered by the Oak Ridge Institute for Science and Education through an interagency agreement between the U.S. Department of Energy and the U.S. Food and Drug Administration.

References

[1] Ken J Beath. “RandomLCA: an R package for latent class with random effects analysis”. In: *J Stat Softw* 81.13 (2017), pp. 1–25.

[2] Andre Esteva et al. “Dermatologist-level classification of skin cancer with deep neural networks”. In: *Nature* 542.7639 (2017), p. 115.

[3] Yinsheng Qu, Ming Tan, and Michael H Kutner. “Random effects models in latent class analysis for evaluating accuracy of diagnostic tests”. In: *Biometrics* (1996), pp. 797–810.