IST Austria: Statistical Machine Learning 2020/21

Christoph Lampert <chl@ist.ac.at>
TAs: Alexandra Peste <alexandra.peste@ist.ac.at>,
Bernd Prach <bernd.prach@ist.ac.at>

Exercise Sheet 2/5 (due date 19/10/2020, 10:15am)
Please send your solutions via email to the TAs

1 Bayes Classifier

In the lecture we saw that the Bayes classifier is

$$c^*(x) := \operatorname{argmax}_{y\in\mathcal{Y}} p(y|x). \tag{1}$$

a) Which of these decision functions is equivalent to c^*? Please give a short argument or derivation why.

- $c_1(x) := \operatorname{argmax}_y p(x)$
- $c_2(x) := \operatorname{argmax}_y p(y)$
- $c_3(x) := \operatorname{argmax}_y p(x, y)$
- $c_4(x) := \operatorname{argmax}_y p(x|y)$

For $\mathcal{Y} = \{-1, +1\}$, we can express the Bayes classifier, e.g., as $c^*(x) = \operatorname{sign}[\log \frac{p(+1|x)}{p(-1|x)}]$

b) Which of the following expressions are equivalent to c^*? No justification is required.

- $c_5(x) := \log[\operatorname{sign}[\frac{p(+1|x)}{p(-1|x)}]]$
- $c_6(x) := \operatorname{sign}[\log p(+1|x) + \log p(-1|x)]$
- $c_7(x) := \operatorname{sign}[\log p(+1|x) - \log p(-1|x)]$
- $c_8(x) := \operatorname{sign}[\log p(x, +1) - \log p(x, -1)]$
- $c_9(x) := \operatorname{sign}[p(+1|x) - p(-1|x)]$
- $c_{10}(x) := \operatorname{sign}[\frac{p(x,+1)}{p(x,-1)} - 1]$
- $c_{11}(x) := \operatorname{sign}[\frac{\log p(+1|x)}{\log p(-1|x)} - 1]$
- $c_{12}(x) := \operatorname{sign}[\log \frac{p(x|+1)}{p(x|-1)} + \log \frac{p(+1)}{p(-1)}]$
- $c_{13}(x) := \begin{cases} +1 & \text{if } p(+1|x) > p(-1|x) \\ -1 & \text{otherwise.} \end{cases}$
- $c_{14}(x) := \operatorname{sign}[\frac{\log(1-p(x,-1))}{\log(1-p(x,+1))}]$

2 Linear Discriminant Analysis (LDA) Classifier

The *Linear Discriminant Analysis (LDA)* classifier is an easy-to-compute method for generative probabilistic classification. For a training set $\mathcal{D} = \{(x^1, y^1), \dots, (x^n, y^n)\} \subset \mathbb{R}^d \times \{1, \dots, M\}$, set

$$\mu := \frac{1}{n}\sum_{i=1}^{n} x^i, \qquad \Sigma := \frac{1}{n}\sum_{i=1}^{n}(x^i - \mu)(x^i - \mu)^\top, \qquad \mu_y := \frac{1}{|\{i : y^i = y\}|}\sum_{\{i:y^i=y\}} x^i, \tag{2}$$

$$\hat{p}(y) = \frac{|\{i : y^i = y\}|}{n} \qquad \hat{p}(x|y) = \frac{1}{\sqrt{2\pi \det \Sigma}} \exp(-\frac{1}{2}(x - \mu_y)^\top \Sigma^{-1} (x - \mu_y)), \qquad \text{for } y \in \mathcal{Y}, \tag{3}$$

a) Show for binary classification ($M = 2$): LDA always leads to a linear decision rule.

b) True or false? The estimate $\hat{p}_{\text{LDA}}(x, y)$ will always converge to the true data distribution $p(x, y)$ for $n \to \infty$.

c) True or false? The resulting decision rule will always converge to the Bayes classifier.

d) Can you come up with a situation (i.e. a data distribution) where *b*) does not hold, but *c*) does?

e) Can you come up with a situation where *b*) does hold, but *c*) does not?

f) Compared to other generative techniques, LDA is popular when there are many classes but only few examples for each class. Can you imagine why?

3 Breaking LDA and LogReg

LogReg and LDA both learn linear decision rules, but usually different ones.
a) Can you construct a data distribution, such that when we sample a dataset from it, Logistic Regression will most likely work quite well, but LDA will fail miserably? (to confirm, you can argue in text or present experiments).
b) Can you do the same but with the roles of LDA and LogReg exchanged?

4 Practical Experiments II

Use again the *wine* dataset from the previous exercise sheet. Train (on the train part of the data) and evaluate (on the test part of the data) the following classifiers from the lecture:

- Linear Discriminant Analysis Classifier
- (Multi-class) Logistic Regression
- As many different multi-class SVMs as you can get your hands on (at least one-versus-rest)

If you rely on existing learning toolboxes, please make sure that you use a plain variant of LogReg without "regularization" or "shrinkage" (or set their strength to 0). For the SVMs, try to find actual *hard-margin* SVMs, and if you can't find any, use a soft-margin one with very large regularization strength, e.g.$C = 1000$.

Please submit your code (in a language of your choice) as well as the resulting error rates.