Statistical Machine Learning

UoC Stats 37700, Winter quarter

Lecture 4: classical linear and quadratic discriminants.

Linear separation

- For two classes in $\mathbb{R}^d$: simple idea: separate the classes using a hyperplane
$$H_{w,b} = \{X : X \cdot w + b \leq 0\}\,;$$
- Simplest extension for several classes: consider a family of linear scores
$$s_y(x) = w_y \cdot x - b_y$$
and the rule
$$f(x) = \operatorname*{Arg\,Max}_{y \in \mathcal{Y}} s_y(x)\,.$$
- Then the separation between any two classes is linear.

Classification via linear regression

- Simplest idea for two classes: perform a standard linear regression of Y (coded e.g. in $\{0,1\}$) by X,

$$\widehat{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$$

  where $\mathbf{X}$ is the $(n, d+1)$ extended data matrix and $\mathbf{Y}$ the $(n,1)$ vector of training classes;
- For a new point x, the linear regression function predicts $x \cdot \widehat{w}$, and the decision function would be $\mathbb{1}\{x \cdot \widehat{w} \geq \frac{1}{2}\}$.

- We can extend this idea to K classes by performing regression on each of the class indicator variables $\mathbb{1}\{Y = y\}, y \in \mathcal{Y}$.
- In matrix form: same as above, replacing $\mathbf{Y}$ by the matrix of indicator responses.
- This is equivalent to solving globally the least squares problem:

$$\min_{\mathbf{W}} \sum_{i=1}^{n} \left\| \overline{Y}_i - X_i \mathbf{W} \right\|^2 ,$$

where $\mathbf{W}$ is a coefficient matrix and $\overline{Y}_i$ is the class indicator vector.

Problems using linear regression

- For multiple classes, a “masking” problem is likely to occur.
- A possible fix is to extend the data vectors with quadratic components.
- Two disadvantages however:
  - masking can still occur when there are many classes.
  - increasing the degree of the components lead to too many parameters and overfitting.

Separating two Gaussians

- We can adopt a simple parametric “generative” approach and model the classes by simple Gaussians.
- Assume we take a Gaussian generative model for the classes distribution:

$$p(x|Y=i) = \frac{1}{\sqrt{(2\pi)^d |\Sigma_i|}} \exp\left(-\frac{1}{2}(x-m_i)^T \Sigma_i^{-1}(x-m_i)\right)$$

- What is the Bayes classifier for this model?
- Remember that if the generating densities for classes 0 and 1 are f_0, f_1 and the marginal probability $p = P(Y=1)$ then the Bayes decision is given by

$$F(x) = \mathrm{Arg\,Max}(pf_1(x), (1-p)f_2(x)) .$$

- Hence, denoting d_i the Mahalanobis distance corresponding to Σ_i ,

$$d_i^2(x,y) = (x-y)^T \Sigma_i^{-1}(x-y)\,,$$

- Then the decision rule is class 1 or 2 depending whether

$$d_1^2(x,m_1) - d_2^2(x,m_2) \le t(p,\Sigma_1,\Sigma_2)\,;$$

  it is a quadratic decision rule (QDA) .
- Case $\Sigma_1 = \Sigma_2$: becomes a linear decision rule (LDA) .
- We can use a pooled estimate for the two covariance matrices:

$$\widehat{\Sigma} = \frac{1}{n-2}\left(S_1 + S_2\right)\,,$$

  where $S_\ell = \sum_{i:Y_i=\ell}(X_i - \widehat{m}_\ell)(X_i - \widehat{m}_\ell)^T$.

- Multiclass: the previous analysis suggests to look at the criterion

$$\operatorname*{Arg\,Min}_{y \in \mathcal{Y}} \delta_y(x)\,,$$

- where

$$\delta_y(x) = \frac{1}{2}(x - m_y)^T \Sigma_y^{-1}(x - m_y) + t_y(p_y, \Sigma_y)\,,$$

  in the general QDA case (then the decision regions are intersections of quadratic regions),
- or

$$\delta_y(x) = -x^T \Sigma^{-1} m_y + t_y(p_y)\,,$$

  in the common variance (LDA) case (then the decision regions are intersections of half-planes)

Relation to linear regression

Theorem

In the two-class case the direction of w found by the Gaussian model coincides with the one found by classical linear regression.

- ...but the constants b differ. In practice it is recommended not to trust either but to consider this as a separate parameter to optimize to reduce the empirical classification error.
- Regression using quadratic terms does not give the same result as QDA.

Fisher's linear discriminant

- Yet another approach to the problem: find the projection maximizing the ratio of inter-class to intra-class variance, for two classes:

$$J(w) = \frac{(w \cdot \widehat{m}_1 - w \cdot \widehat{m}_2)^2}{w^T(S_1 + S_2)w},$$

  where $\widehat{m}_\ell$ are the empirical class means and $S_\ell = \sum_{i:Y_i=\ell}(X_i - \widehat{m}_\ell)(X_i - \widehat{m}_\ell)^T$
- Finding $\frac{dJ}{dw} = 0$ leads to the solution

$$w = \lambda(S_1 + S_2)^{-1}(\widehat{m}_1 - \widehat{m}_2)$$

  (again, the scaling is arbitrary).
- The projection direction coincides with the previous methods; Fisher's criterion only provides the projection direction (again, optimize the constant separately)

Fisher's discriminant in multi-class

- Fisher's criterion can be extended to the multi-class case by maximizing the ratio (Rayleigh coefficient)

$$J(w) = \frac{w^T M w}{w^T S w}$$

  where $S = \sum_y S_y$ is the pooled intraclass covariance and $M = \sum_y (m_y - m)(m_y - m)^T$ is the interclass covariance (covariance of the class centroids).
- (Note that normalization of the matrices is unimportant)
- Leads to the generalized eigenvalue problem

$$Mw = \lambda S w$$

- Can be iterated to find $|\mathcal{Y}| - 1$ dimensions by constraining orthogonality (for the scalar product $\langle w, w' \rangle = w^T S w'$) with previously found directions.
- Equivalent to the following: "whiten" the data by applying $S^{-\frac{1}{2}}$; perform PCA on the transformed class centroids; apply $S^{-\frac{1}{2}}$ to the found directions.

Properties of Fisher's canonical projections

- This is a linear dimension reduction method aimed at "separating" the classes (using 1st and 2nd moment information only).
- Invariant by any linear transform of the input space.
- When we take $L = \min(d, |\mathcal{Y}| - 1)$ canonical coordinates, this "commutes" with LDA.
- When we take $L < min(d, |\mathcal{Y}| - 1)$ canonical coordinates, this is equivalent to a reduced rank LDA, i.e. where we require the mean of the Gaussians in the model to belong to a space of dimension L (and perform ML fitting).
- It can also be seen as a CCA of X wrt. the class indicator function $\overline{Y}$.

Regularized linear and quadratic discriminant

- When the dimension d is too large, overfitting and instability can occur.
- Looking back at standard linear regression, a possible is ridge regression finding

$$\widehat{\beta}_\lambda = \operatorname*{Arg\,Min}_{\beta} \left(\sum_{i=1}^{N} \left(Y_i - \beta_0 - \sum_{j=1}^{d} x_{ij}\beta_j \right)^2 + \lambda \sum_{i=1}^{p} \beta_j^2 \right) ;$$

- The solution is given by

$$\widehat{\beta}_{1 \le i \le d} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}$$

"regularization by shrinkage".

- By a (weak) analogy with ridge regression we can consider the following regularized version for the covariance estimation in LDA:

$$\widehat{\Sigma}_\gamma = \gamma\widehat{\Sigma} + (1-\gamma)\widehat{\sigma}^2\mathbf{I}$$

  another possibility is

$$\widehat{\Sigma}_\gamma = \gamma\widehat{\Sigma} + (1-\gamma)\mathbf{D}\,,$$

  where $\mathbf{D}$ is the diagonal matrix formed with entries $\widehat{\sigma_i}^2$.
- We can also regularize QDA using the following scheme for the estimator of the covariance matrix for class k:

$$\widehat{\Sigma}_k(\alpha) = \alpha\widehat{\Sigma}_k + (1-\alpha)\widehat{\Sigma}$$

- ...we can even combine the two.
- In practice, as usual it is recommended to use cross-validation to tune the parameters.

Linear Logistic regression

- Recall that in the 2-class case logistic regression aims at finding the log-odds ratio function

$$\log \frac{P(Y=1|X=x)}{P(Y=0|X=x)} = \log \frac{\eta(x)}{1-\eta(x)} ;$$

- In the multiclass case, this can be generalized to log-odds ratio wrt. some (arbitrary) reference class:

$$s_i(x) = \log \frac{P(Y=i|X=x)}{P(Y=0|X=x)} ;$$

  again the resulting (plug-in) classifier outputs the max of the "score functions".
- If we model scores by linear functions, we get again a linear classifier.

- The model
$$s_i(x) = \beta_{i0} + \beta_i \cdot x$$
gives rise to the conditional class probabilities
$$P(Y = i | X = x) = \frac{\exp\left(\beta_{i0} + \beta_i \cdot x\right)}{1 + \sum_{\ell=1}^{K-1} \exp\left(\beta_{\ell 0} + \beta_\ell \cdot x\right)},$$
we can fit this using Maximum Likelihood.

Algorithm for linear logistic regression

- We consider the 2-class case ($\mathcal{Y} = \{0, 1\}$).
- The log-likelihood function is

$$\ell(\beta) = \sum_{i=1}^{n} \left(Y_i \beta \cdot X_i - \log(1 + \exp(\beta \cdot X_i)) \right) ;$$

(where the data points X_i are augmented with a contant coordinate) and

$$\frac{d\ell}{d\beta} = \sum_{i=1}^{n} X_i \left(Y_i - \eta(X_i, \beta) \right) \qquad (= 0) ;$$

we can solve this using a Newton-Raphson algorithm with step

$$\widehat{\beta}^{new} = \widehat{\beta}^{old} - \left(\frac{d^2\ell}{d\beta d\beta^T} \right)^{-1} \frac{d\ell}{d\beta} .$$

- We have

$$\left(\frac{d^2\ell}{d\beta d\beta^T}\right) = -\sum_{i=1}^{n} X_i X_i^T \eta(X_i, \beta)(1 - \eta(X_i, \beta))\,;$$

- If we denote $\mathbf{W}$ the diagonal matrix of weights $\eta(X_i, \beta)(1 - \eta(X_i, \beta))$, we can rewrite the NR step in matrix form as

$$\widehat{\beta}^{new} = (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W} \mathbf{Z}\,,$$

where

$$\mathbf{Z} = \mathbf{X}\widehat{\beta}^{old} + \mathbf{W}^{-1}(\mathbf{Y} - \overline{\eta})\,.$$

- This can be seen as an iterated modified least squares fitting.

LDA vs. logistic regression

- LDA and logistic regression both fit a linear model to the log-odds ratio.
- They do not result in the same output however. Why?
- The answer is that logistic regression only fits the conditional densities $P(Y = i|X)$ and remains "agnostic" as to the distribution of covariate X. LDA on the other hand implicitly fits a distribution for the joint distribution $P(Y|X)$ (mixture of Gaussians).
- In practice, logistic regression is therefore considered more adaptive, but also less robust.

The linear perceptron

- Assume that the “point clouds” for two the classes in the training set turn out to be perfectly separable by some hyperplane.
- Then LDA will not necessarily return a hyperplane having zero training error.
- On the other hand, logistic regression will return infinite parameters (why?)
- Other approach: consider minimizing a criterion based on the distance of misclassified examples to the hyperplane:

$$D(w,b) = -\sum_{i:Y_i(X_i\cdot w+b)<0} Y_i\,(X_i \cdot w + b)\ .$$

  (here we assume $Y_i \in \{-1,1\}$!). (Note that actually the average distance to the hyperplane would be $\|w\|^{-1} D(w,b)$.)
- Principle of perceptron training (more or less): minimize the above by a kind of stochastic gradient descent.

Convergence of perceptron training

- Very simple iterative rule: first, put $R = \max_i \|X_i\|$.
  - If all points are correctly classified, stop.
  - If there are misclassified points, choose such a point (X_i, Y_i) *arbitrarily*.
  - Put
$$\begin{bmatrix} w^{new} \\ b^{new} \end{bmatrix} = \begin{bmatrix} w^{old} \\ b^{old} \end{bmatrix} + \begin{bmatrix} Y_i X_i \\ Y_i R^2 \end{bmatrix} .$$
  - Repeat.

Theorem

If there exists (w^*, b^*) a separating hyperplane, such that for all i

$$Y_i \left(w^* \cdot X_i + b^*\right) \geq \gamma ,$$

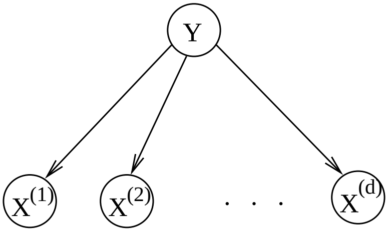
then above algorithm will eventually find a separating hyperplane in a finite number of steps bounded by $\left(\frac{2R}{\gamma}\right)^2$.

Some problems with the perceptron algorithm:

- The number of steps required to converge can be large!
- If the classes are not separable, there is no guarantee of convergence. In fact, cycles can occur.
- There is no regularization and so no protection against overfitting (the number of steps can be used as regularization though)

The "naive Bayes" classifier

- Assume that X is a vector of binary features (possibly in very high dimension, $X \in \{0,1\}^d$.
- We don't want to model very complicated dependencies. Naive assumption: given Y, the coordinates of X are independent!
- Can be seen as an elementary graphical model:



- Assume we know the probability distribution of each coordinate (a Bernoulli variable) given Y, $p_{k,-1} = P(X^{(k)} = 1|Y = -1)$; $p_{i,1} = P(X^{(k)} = 1|Y = 1)$.
- In this case the Bayes classifier is given by the sign of

$$\log \frac{P(Y = 1|X)}{P(Y = -1|X)} = \sum_k X^{(k)} \alpha_k + a$$

where

$$\alpha_k = \log \frac{p_{k,1}(1 - p_{k,-1})}{(1 - p_{k,1})p_{k,-1}}; \quad a = \sum_k \log \frac{1 - p_{k,1}}{1 - p_{k,-1}} + \log \frac{P(Y = 1)}{P(Y = -1)};$$

- In practice: like for LDA, it is recommended to optimize the constant a separately (to minimize the training error).

Naive Bayes classifier generalized

- We can generalize this idea to continuous-valued coordinates with the same conditional independence hypothesis.
- We estimate the conditional density of each coordinate (e.g. with a Gaussian)
- The decision function becomes an additive model:

$$\widehat{f}(x) = \mathrm{sign}\left(\sum_k \widehat{f}^{(k)}(x^{(k)})\right)$$

- Advantage of naive Bayes: robust also in high dimension, simple and surprisingly good in a number of situations even when the assumption obviously does not hold.
- Disadvantage: generally does not match the performance of more flexible methods.