Machine Learning Element 1 Project

Read carefully

- Each pair of students must choose data-sets and a couple (Coding Design, Decoding Design) differently from other pairs.
- The deliverable is a compressed file `name_student1_name_student2.rar` or `name_student1_name_student2.zip`.
- Each approach is in a separate `.py` or `.ipynb` file: Example: `one_vs_one_perceptron.ipynb`
- Put `package_name==package_version` in file `requirements.txt`, in the case of using a package to install.
- **Last deadline for receipt of projects is ....**

Multiclass Classification

1. Choose three multiclass classification datasets: the first one is with linearly separable classes, the second one is like the first but with a noise, and the third one is not linearly separable.
2. Visualize data.
3. For each dataset Apply these algorithms: Perceptron, Pocket, Adaline, Logistic regression and Polynomial transformation with Pocket and Adaline.
   For each algorithm:
   (a) Apply the three multiclass classification methods: One-vs-all, one-vs-one and Error correcting codes by selecting a couple (Coding Design, Decoding Design) in the lists below, or an other approache that have not been mentioned in the lists.
   (b) Compare all the results obtained and indicate the best approach.

List 1: Coding Designs

- Dense Random (Allwein et al., 2002): $n = 10 \cdot \log N_c$ dichotomizers are suggested to be learnt for N_c classes, where $P(-1) = 1 - P(+1)$, being $P(-1)$ and $P(+1)$ the probability of the symbols -1 and $+1$ to appear, respectively. Then, from a set of defined random matrices, the one which maximizes a decoding measure among all possible rows of M_c is selected.
- Sparse Random (Escalera et al., 2009): $n = 15 \cdot \log N_C$ dichotomizers are suggested to be learnt for N_c classes, where $P(0) = 1 - P(-1) - P(+1)$, defining a set of random matrices M_c and selecting the one which maximizes a decoding measure among all possible rows of M_c.
- DECOC (Pujol et al., 2006): problem-dependent design that uses $n = N_c - 1$ dichotomizers. The partitions of the problem are learnt by means of a binary tree structure using exhaustive search or a SFFS criterion. Finally, each internal node of the tree is embedded as a column in M_c.
- Forest-ECOC (Escalera et al., 2007): problem-dependent design that uses $n = (N_c - 1) \cdot T$ dichotomizers, where T stands for the number of binary tree structures to be embedded. This approach extends the variability of the classifiers of the DECOC design by including extra dichotomizers.
- ECOC-ONE (Pujol et al., 2008): problem-dependent design that uses $n = 2 \cdot N_c$ suggested dichotomizers. A validation sub-set is used to extend any initial matrix M_c and to increase its generalization by including new dichotomizers that focus on difficult to split classes.

List 2: Decoding Designs

- Hamming decoding: $\mathrm{HD}(x, y_i) = \sum_{j=1}^{n} \left(1 - \mathrm{sign}\left(x^j \cdot y_i^j\right)\right)/2$, being x a test codeword and y_i a codeword from M_C corresponding to class C_i.
- Inverse Hamming decoding: $\mathrm{IHD}(x, y_i) = \max\left(\Delta^{-1} D^T\right)$, where $\Delta(i_1, i_2) = HD(y_{i_1}, y_{i_2})$, and D is the vector of Hamming decoding values of the test codeword x for each of the base codewords y_i.
- Euclidean decoding: $ED(x, y_i) = \sqrt{\sum_{j=1}^{n}\left(x^j - y_i^j\right)^2}$.
- Attenuated Euclidean decoding: $AED(x, y_i) = \sqrt{\sum_{j=1}^{n}\left|y_i^j\|x^j\right|\left(x^j - y_i^j\right)^2}$.
- Loss-based decoding: $LB(\rho, y_i) = \sum_{j=1}^{n} L\left(y_i^j \cdot f^j(\rho)\right)$, where ρ is a test sample, L is a lossfunction, and f is a real-valued function $f: R^n \rightarrow \mathcal{R}$.
- Probabilistic-based decoding:
  $PD(y_i, x) = -\log\left(\Pi_{j\in[1,\dots,n]:M_c(i,j)\neq 0} P\left(x^j = M_c(i,j) \mid f^j\right) + K\right)$, where K is a constant factor that collects the probability mass dispersed on the invalid codes, and the probability $P\left(x^j = M_c(i,j) \mid f^j\right)$ is estimated by means of $P\left(x^j = y_i^j \mid f^j\right) = \frac{1}{1+\mathrm{e}_i^{v_i^j\left(v^j f^j + \omega^j\right)}}$, where vectors U and ω are obtained by solving an optimization problem (Passerini et al., 2004).
- Laplacian decoding: $\mathrm{LAP}(x, y_i) = \frac{\alpha_i + 1}{\alpha_i + \beta_i + K}$, where α_i is the number of matched positions between x and y_i, β_i is the number of miss-matches without considering the positions coded by 0 , and K is an integer value that codifies the number of classes considered by the classifier.
- Pessimistic β-Density Distribution decoding: accuracy s_i : $\int_{\nu_i - s_i}^{\nu_i} \Psi_i(\nu, \alpha_i, \beta_i)\, d\nu = \frac{1}{3}$, where $\Psi_i(\nu, \alpha_i, \beta_i) = \frac{1}{K}\nu^{\alpha_i}(1-\nu)^{\beta_i}$, Ψ_i is the β-Density Distribution between a codeword x and a class codeword y_i for class c_i, and $\nu \in \mathcal{R} : [0, 1]$.
- Loss-Weighted decoding: $LW(\rho, i) = \sum_{j=1}^{n} M_W(i,j) L\left(y_i^j \cdot f(\rho, j)\right)$, where $M_W(i,j) = \frac{H(i,j)}{\sum_{j=1}^{n} H(i,j)}$, $H(i,j) = \frac{1}{m_i}\sum_{k=1}^{m_i} \phi\left(h^j\left(\rho_k^i\right), i, j\right)$, $\phi\left(x^j, i, j\right) = \begin{cases} 1, & \text{if} \quad x^j = y_i^j, \\ 0, & \text{otherwise.} \end{cases}$, m_i is the number of training samples from class C_i, and ρ_k^i is the k th sample from class C_i.

Good luck

References

E. Allwein, R. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. Journal of Machine Learning Research, 1:113–141, 2002.

T. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research, 2:263–282, 1995.

S. Escalera, Oriol Pujol, and Petia Radeva. Boosted landmarks of contextual descriptors and Forest- ECOC: A novel framework to detect and classify objects in clutter scenes. Pattern Recognition Letters, 28(13):1759–1768, 2007.

S. Escalera, O. Pujol, and P. Radeva. On the decoding process in ternary error-correcting output codes. IEEE Transactions in Pattern Analysis and Machine Intelligence, 99, 2008.

S. Escalera, O. Pujol, and P. Radeva. Separability of ternary codes for sparse designs of error- correcting output codes. Pattern Recognition Letters, 30:285–297, 2009.

E. B. Kong and T. G. Dietterich. Error-correcting output coding corrects bias and variance. Inter- national Conference of Machine Learning, pages 313–321, 1995.

N. J. Nilsson. Learning Machines. McGraw-Hill, 1965.

A. Passerini, M. Pontil, and P. Frasconi. New results on error correcting output codes of kernel machines. IEEE Transactions on Neural Networks, 15(1):45–54, 2004.

O. Pujol, P. Radeva, , and J. Vitrià. Discriminant ECOC: A heuristic method for application depen- dent design of error correcting output codes. IEEE Transactions in Pattern Analysis and Machine Intelligence, 28:1001–1007, 2006.

O. Pujol, S. Escalera, and P. Radeva. An incremental node embedding technique for error-correcting output codes. Pattern Recognition, 4:713–725, 2008.

R. Rifkin and A. Klautau. In defense of one-vs-all classification. The Journal of Machine Learning Research, 5:101–141, 2004